深度学习-图像视频处理

史春奇

2017 年

目录

第三部分-图像视频处理

- 深度学习图像视频处理
  - 谁最早开始引入深度学习？
  - 如何引入深度学习？
  - 单纯引入深度学习有哪些局限性？
- 深度学习图像处理模块
  - RPN 模块是如何取代 Selective Search 的？
  - FPN 模块是如何取代 HoG Pyramid 的？
  - Box Regression 是如何融合的？

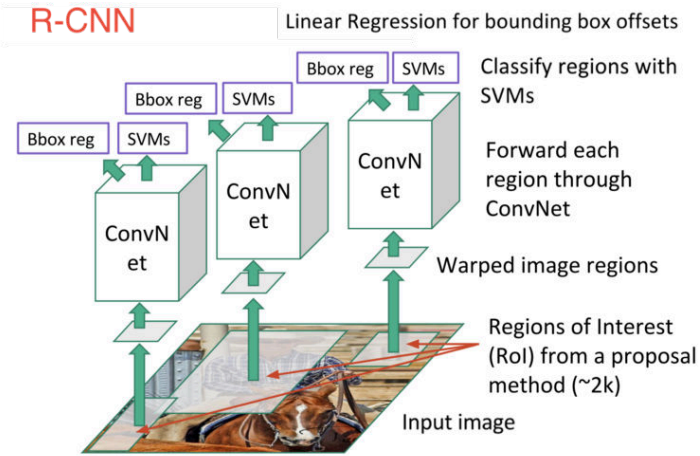
1 图像视频处理简介

1.1 计算视觉常见应用

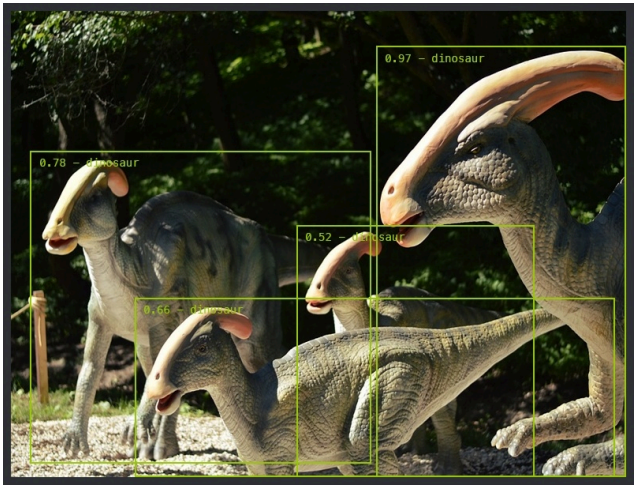
- 分类

- 定位

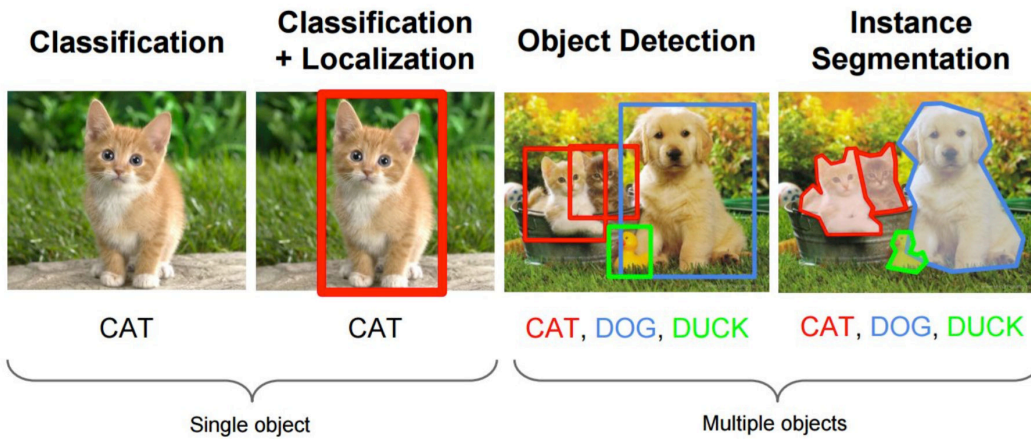
- 分割


R-CNN
Linear Regression for bounding box offsets
Bbox reg
SVMs
Classify regions with SVMs
Forward each region through ConvNet
ConvNet
Warped image regions
Regions of Interest (RoI) from a proposal method (~2k)
Input image

- 物体检测

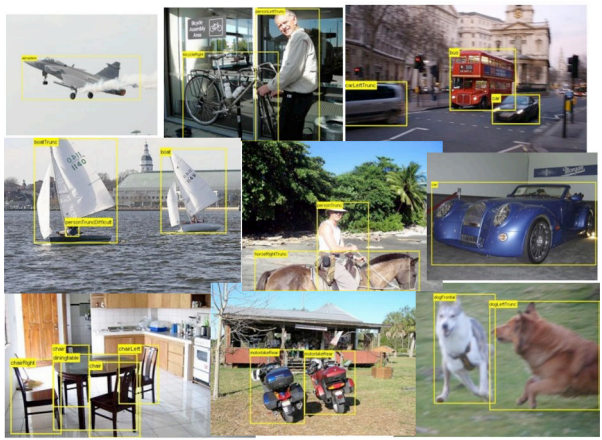


- 区分分类、定位、识别、分割

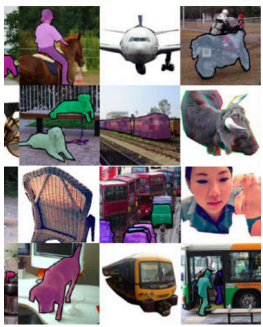

Classification
Classification + Localization
Object Detection
Instance Segmentation
CAT
CAT
CAT, DOG, DUCK
CAT, DOG, DUCK
Single object
Multiple objects

1.2 图像处理数据集

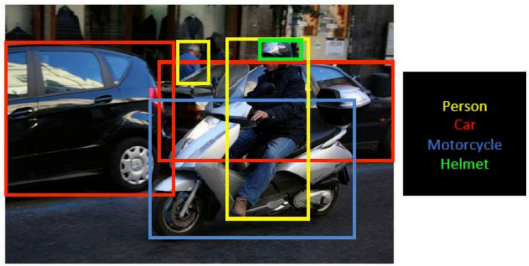
- PASCAL Visual Object Classes (VOC) 挑战
  - 人、车、自行车、公交车、飞机、羊、牛、桌等 20 大类



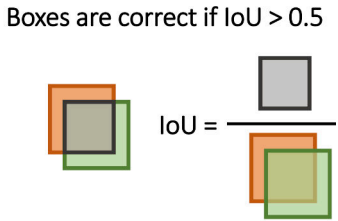
- MS COCO: Microsoft Common Object in Context
  - 80 大类，多目标

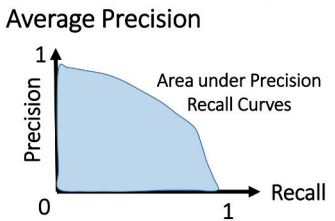


- ImageNet Object Dtection: ILSVRC DET 任务
  - 200 类别，578,482 图片

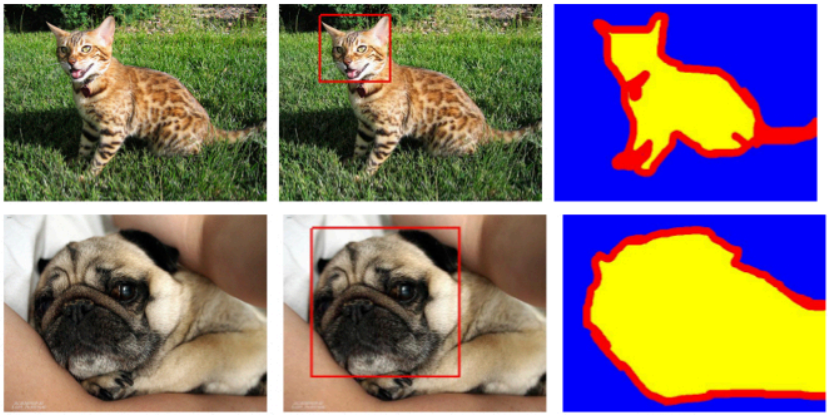


This year: 5,500 new test images with bounding boxes fully annotated

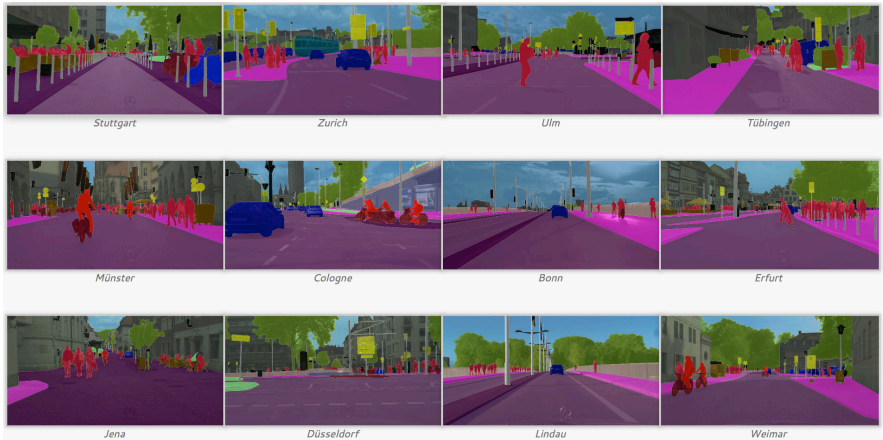




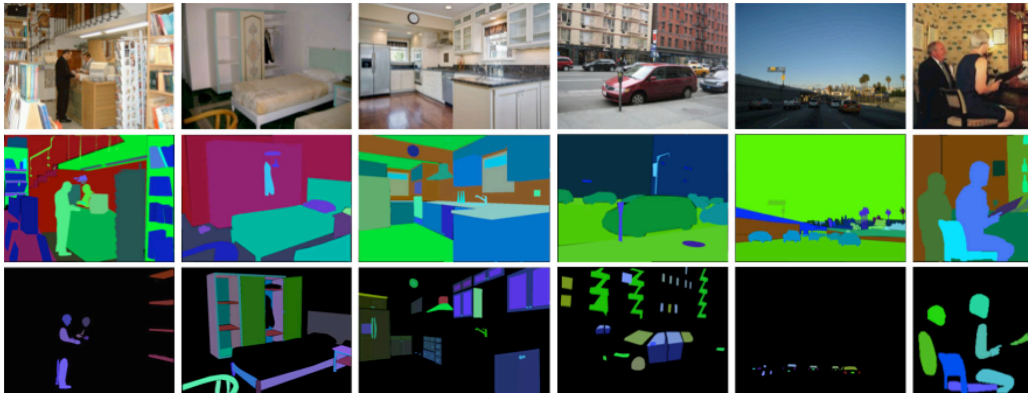
- Oxford-IIIT Pet Dataset
  - 37 类别，每个类别 200 图片



- Cityscapes Dataset
  - 30 类别，25,000 + 真实开车场景图片



- ADE20K Dataset
  - 150+ 类别，22,000 + 普通场景图片



1.3 图像处理重要牛人

- Navneet Dalal 和 Bill Triggs (INRIA)
  - 2005 CVPR, 将 Histogram of Gradient (HOG) 特征应用到图像

Navneet Dalal

Bill Triggs

  - Conference on Computer Vision and Pattern Recognition (CVPR)
  - French National Institute for Research in Computer Science and Automation

(INRIA)

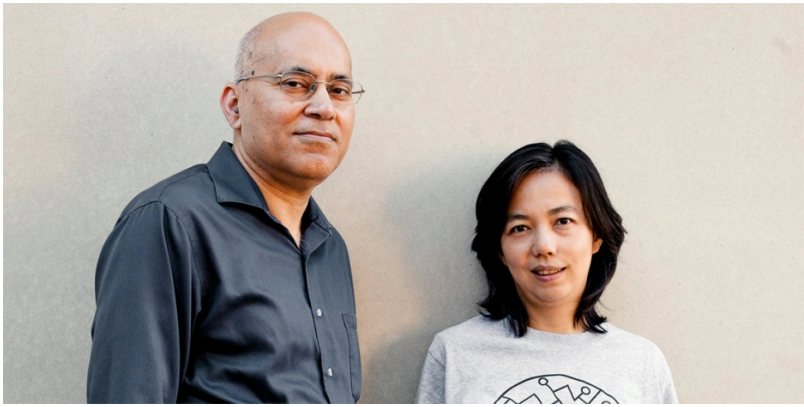
– 2015 Longuet-Higgins Prize (Sponsored by Microsoft Research)

- Pedro Felipe Felzenszwalb
  - 2004 年提出过分割算法
  - 和学生 Ross Girshick 一起发明了 DPM，深化了 HOG

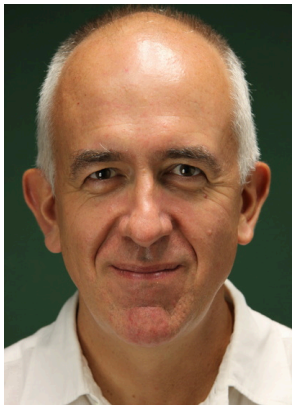
- Koen van de Sande
  - 提出 Selective Search

- Ross Girshick (FAIR)
  - RBG 大神、发明了 R-CNN、Mask R-CNN

- Jitendra Malik
  - RBG 大神的博士后导师，R-CNN 发明人
  - 鼓励 RBG 引入 CNN 深度学习



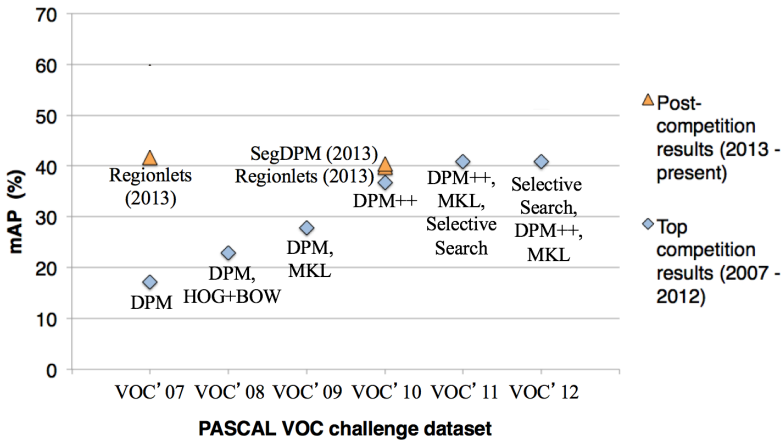
- Pietro Perona
  - 和 Piotr Dollar 一起提出 Fast Feature Pyramids



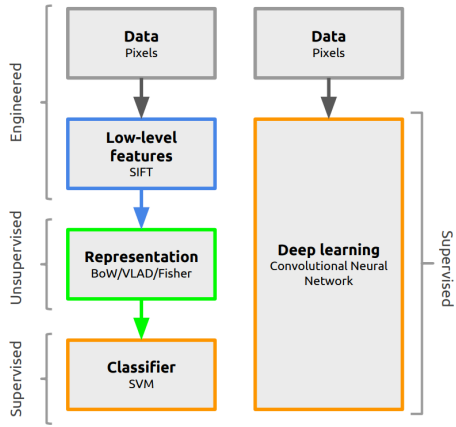
- 何凯明 (FAIR)
  - ResNet，SPP，Mask-CNN 发明人

1.4 图像处理背景

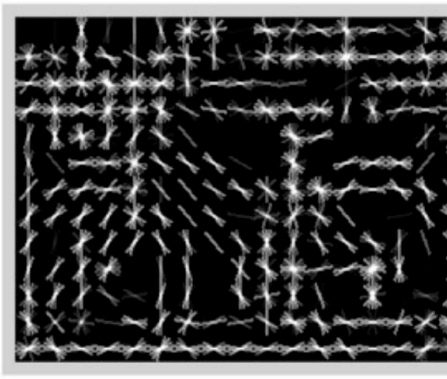
- 2012 年之前的物体识别
  - 2013 年 AlexNet 很空出世！



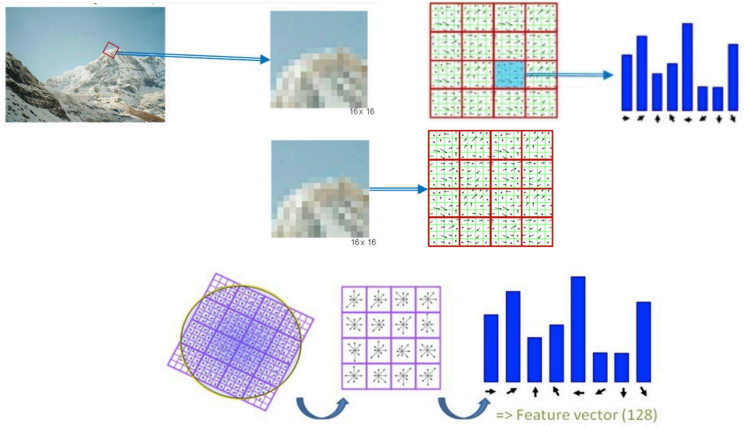
- 开启转型之路
  - 深度网络是如何取代不同功能模块的？



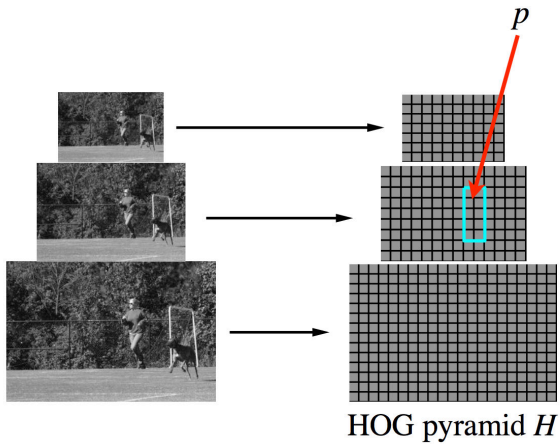
- Histogram of Gradient (HOG) 特征
  - 8x8 像素框内计算方向梯度直方图



- HoG: SIFT



- HoG 过滤器
  - 点积得分最大、每张图 250,000 个位置 (p)、目标稀疏

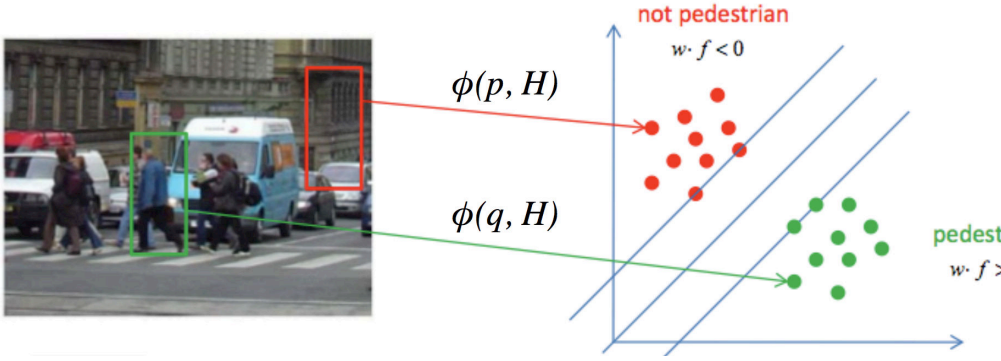


filter F

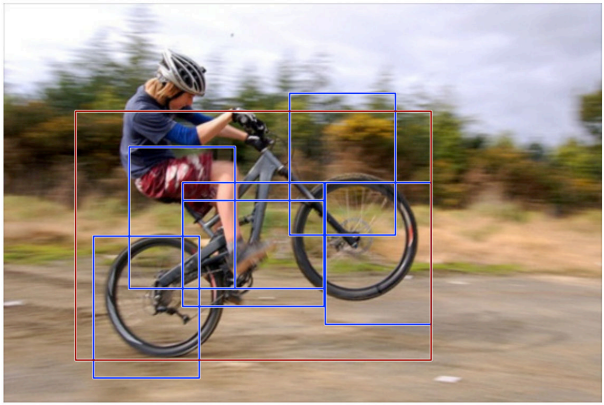
Score of F at position p is $F \cdot \phi(p, H)$

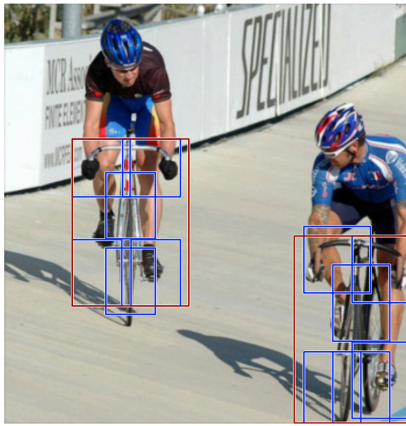
$\phi(p, H)$ = HOG features in subwindow specified by p

- HoG 特征的 SVM 分类
  - 非平衡数据 (背景 VS 目标)，SVM 优势明显

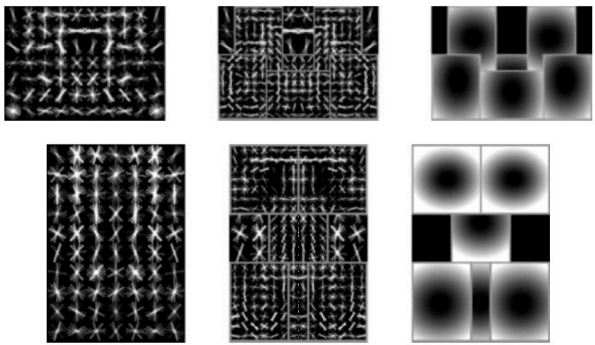


- 可变形组件模型 Deformable Part Model
  - 定义一组固定比例的模版 (整体模版、局部模版)





- 自行车的 DPM
  - 组件之间可以变形



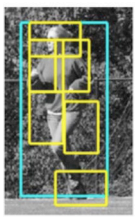
- DPM 的得分 = 模版匹配得分 - 弹性代价
  - 各个过滤器的匹配得分减去组件间距离代价

$$\text{score}(p_0,\ldots,p_n) = \underbrace{\sum_{i=0}^{n} F_i \cdot \phi(H,p_i)}_{\text{“data term”}} - \underbrace{\sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)}_{\text{“spatial prior”}}$$

filters

displacements

deformation parameters

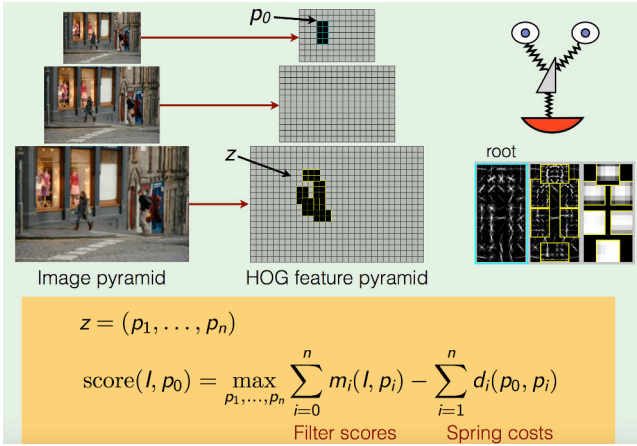


$$\text{score}(z) = \beta \cdot \Psi(H,z)$$

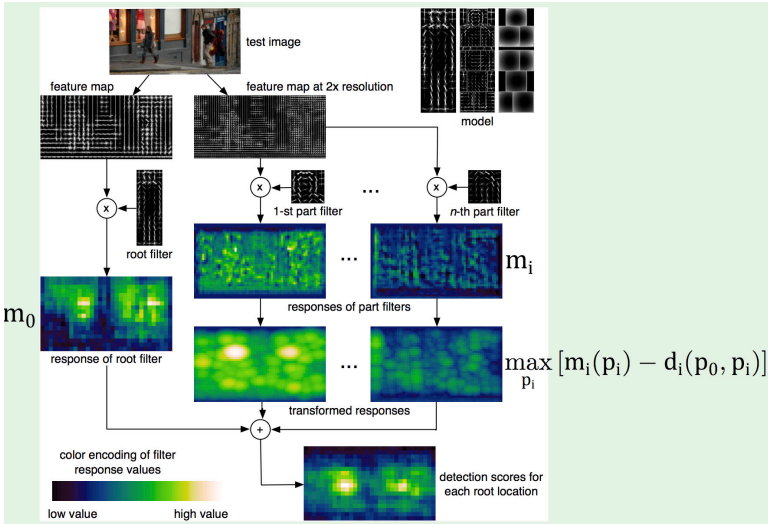
concatenation filters and deformation parameters

concatenation of HOG features and part displacement features

- DPM 的最佳得分
  - 不同组件组合的最佳得分

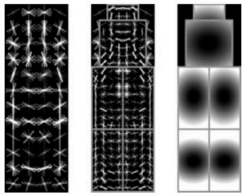


- DPM 得分计算流程

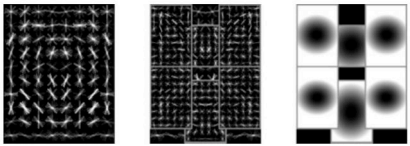

test image
feature map
feature map at 2x resolution
model
1-st part filter
n-th part filter
root filter
responses of part filters
response of root filter
transformed responses
color encoding of filter response values
low value
high value
detection scores for each root location
m_0
m_i
$\max_{p_i} [m_i(p_i) - d_i(p_0, p_i)]$

- 不同物体的 DPM 组件

Bottle

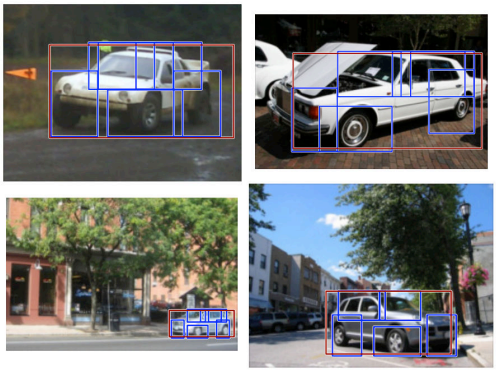


Cat

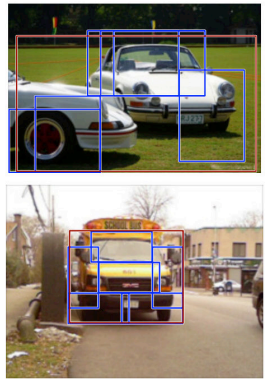


- DPM 失效情况

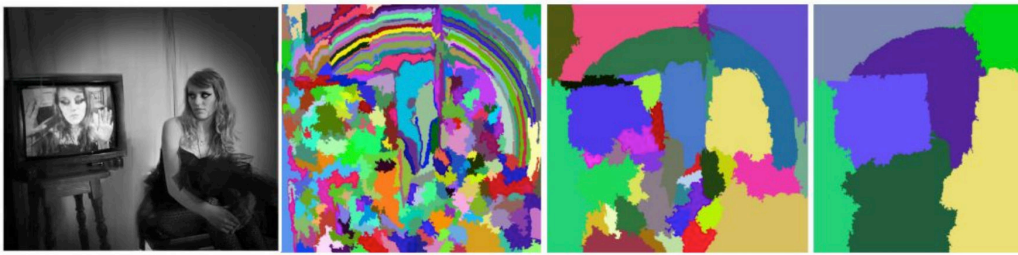
high scoring true positives



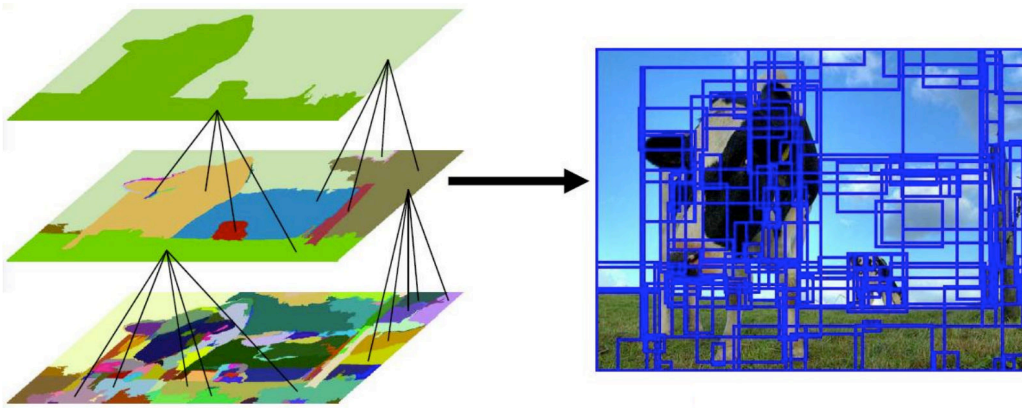
high scoring false positives



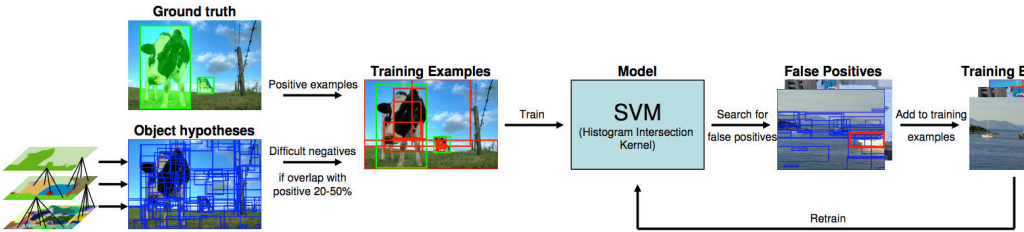
- 过分割：Over Segmentation
  - 过分割后基于颜色纹理等相似度合并



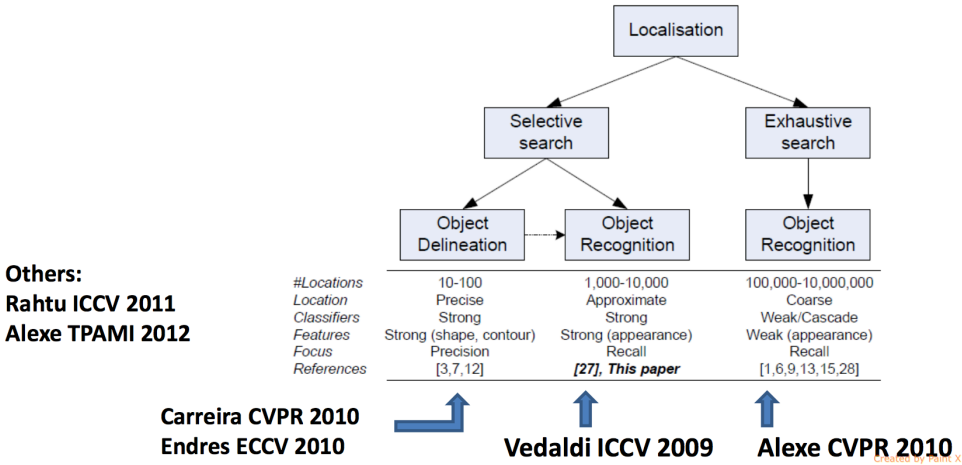
- Selective Search 思想
  - 过分割、分层合并、建议区域排序



- 基于 Selective Search 的物体识别
  - SS 提供区域建议、再通过 HOG 确认
  - 其他建议区域可以作为负样本

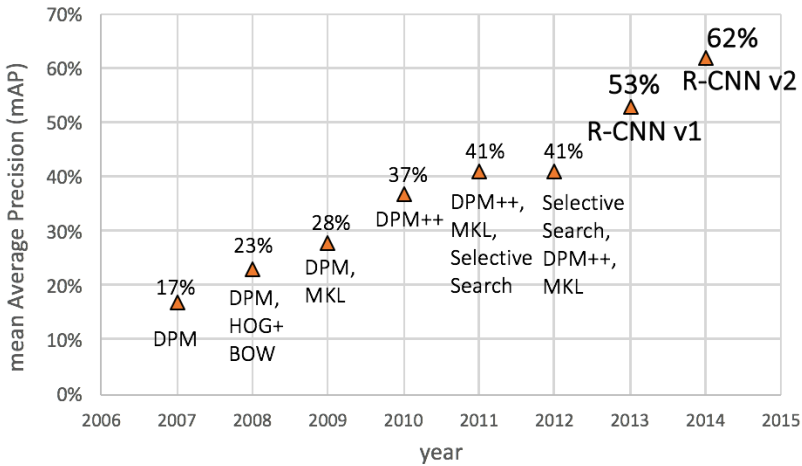


- Selective Search 优势对比



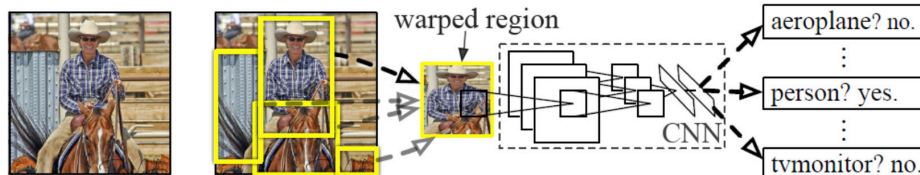
2 图像视频处理模型

- R-CNN 的出现



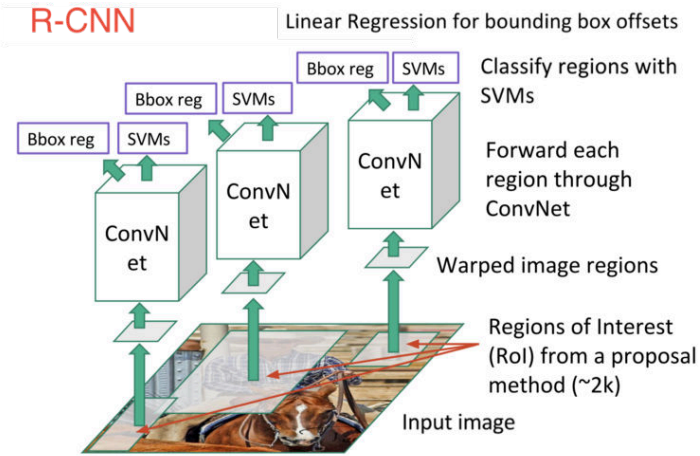
2.1 R-CNN

- R-CNN 本质
  - 用 CNN 取代 HOG、DPM 等

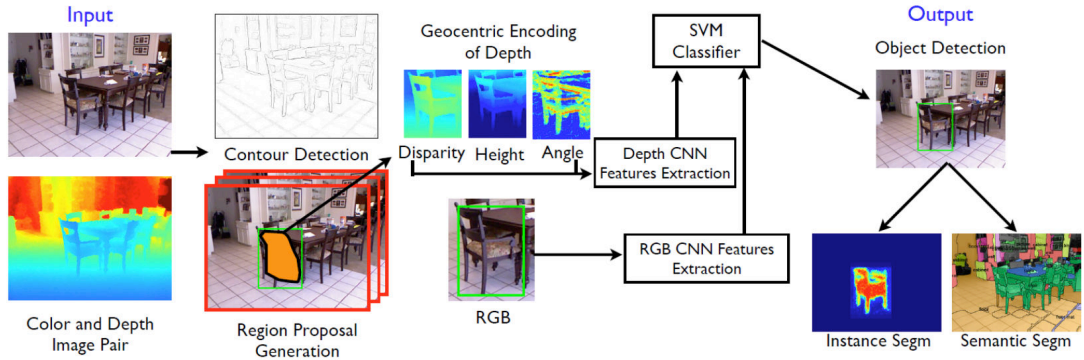


	localization	feature extraction	classification
this paper:	selective search	deep learning CNN	binary linear SVM
alternatives:	objectness, constrained parametric min-cuts, sliding window ...	HOG, SIFT, LBP, BoW, DPM ...	SVM, Neural networks, Logistic regression ...

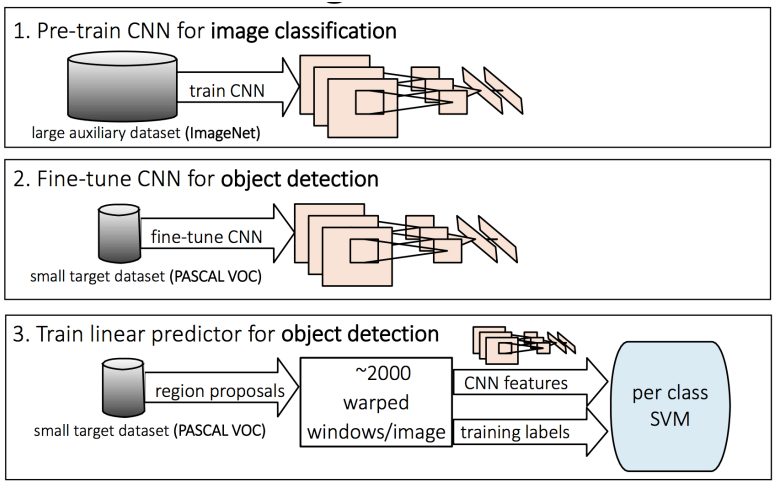
- R-CNN 结构


R-CNN
Linear Regression for bounding box offsets
Bbox reg
SVMs
Classify regions with SVMs
Forward each region through ConvNet
ConvNet
Warped image regions
Regions of Interest (RoI) from a proposal method (~2k)
Input image

- R-CNN 多 CNN 结构


Input
Color and Depth
Image Pair
Contour Detection
Region Proposal
Generation
Geocentric Encoding
of Depth
Disparity
Height
Angle
Depth CNN
Features Extraction
RGB
RGB CNN Features
Extraction
SVM
Classifier
Output
Object Detection
Instance Segm
Semantic Segm

- R-CNN 训练
  - 通过分类预训练

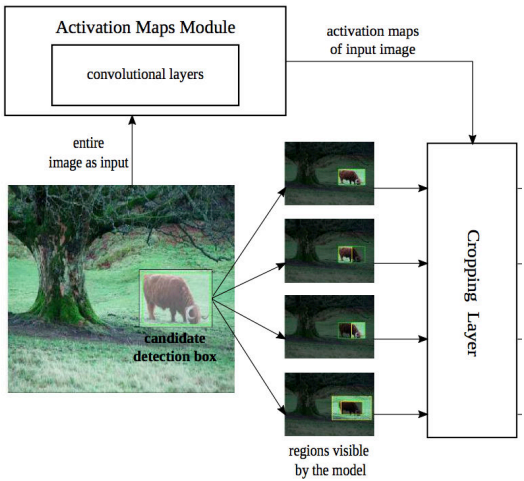


- R-CNN 小结

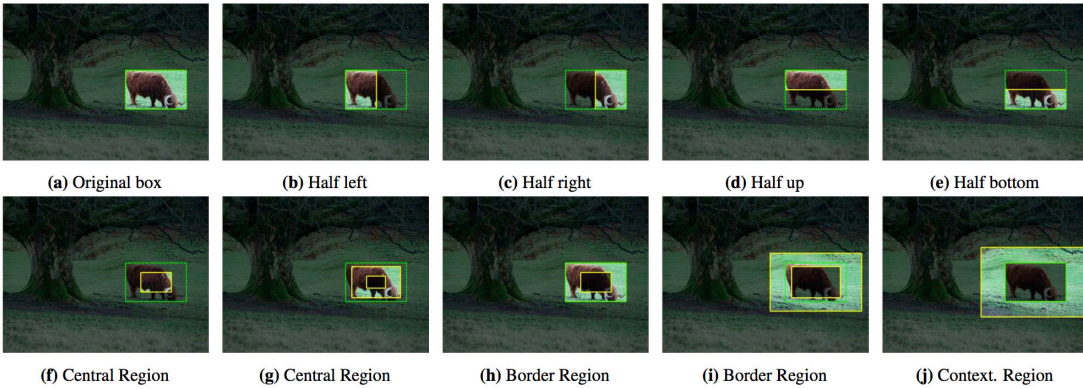
1. 基于 Selective Search 的方法
2. 用 CNN 取代了 HOG、DPM 做特征提取
3. 分类依然使用 SVM
4. 不是端到端的模型
5. 速度相当的慢，流程很麻烦

2.2 MR-CNN

- MR-CNN (Multi-Region CNN)



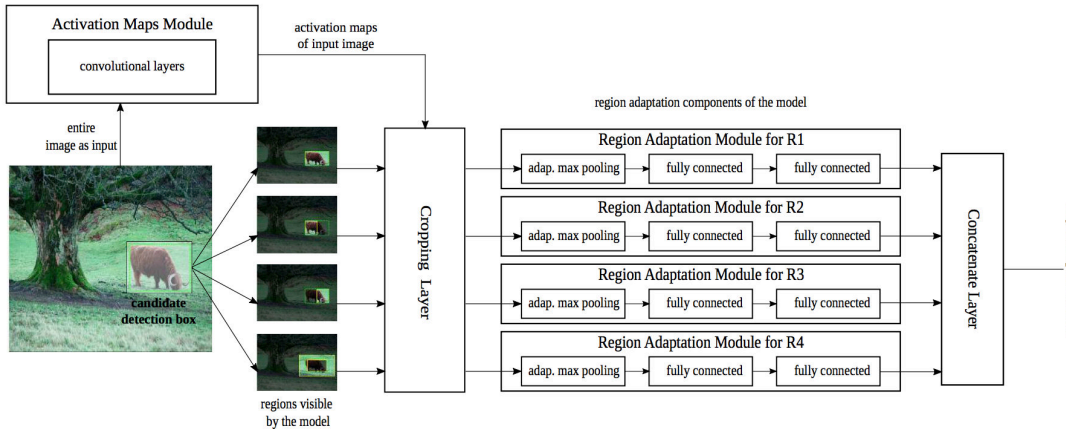
- Multi-Region



(a) Original box **(b)** Half left **(c)** Half right **(d)** Half up **(e)** Half bottom

(f) Central Region **(g)** Central Region **(h)** Border Region **(i)** Border Region **(j)** Context. Region

- MR-CNN 架构

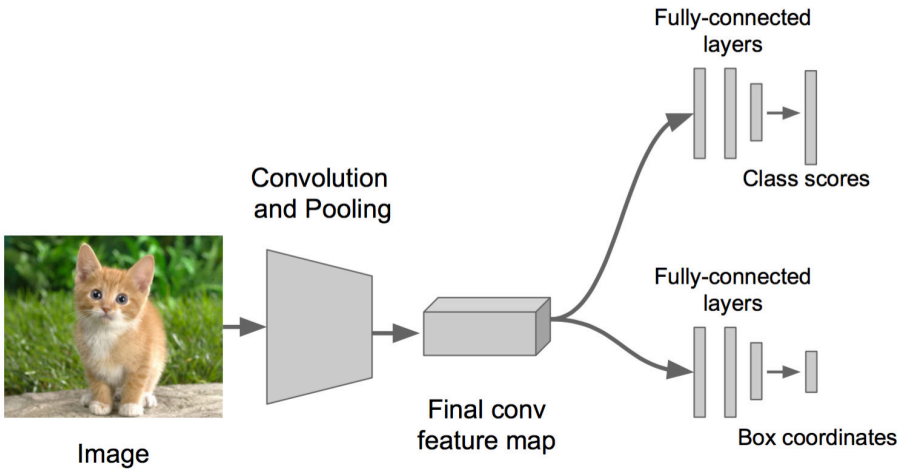

Activation Maps Module
convolutional layers
activation maps of input image
entire image as input
candidate detection box
regions visible by the model
Cropping Layer
region adaptation components of the model
Region Adaptation Module for R1
adap. max pooling
fully connected
fully connected
Region Adaptation Module for R2
adap. max pooling
fully connected
fully connected
Region Adaptation Module for R3
adap. max pooling
fully connected
fully connected
Region Adaptation Module for R4
adap. max pooling
fully connected
fully connected
Concatenate Layer
Object Representation

- MR-CNN 小结

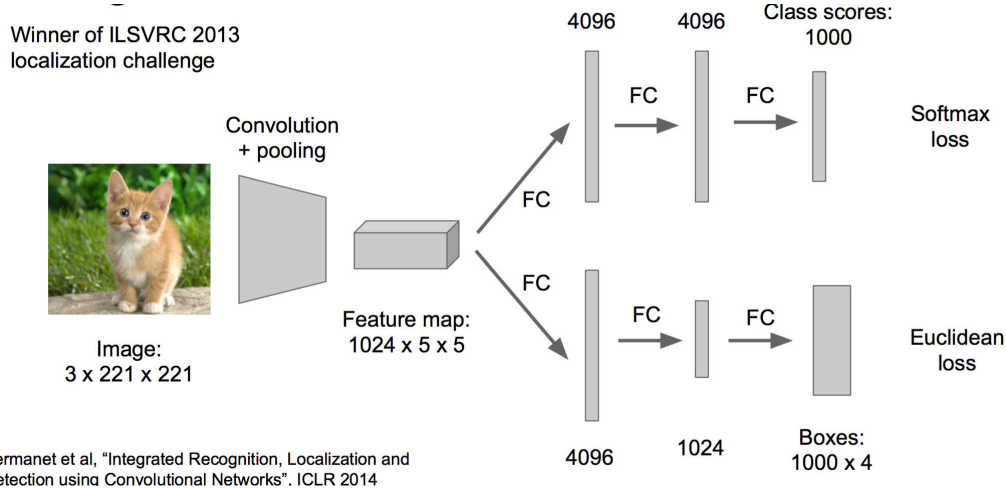
1. 基于 R-CNN 框架，引入多区域框架
2. 准确率比 R-CNN 有较大提高，对部分重叠交叉的情况改善
3. 缺陷和 R-CNN 类似

2.3 OverFeat

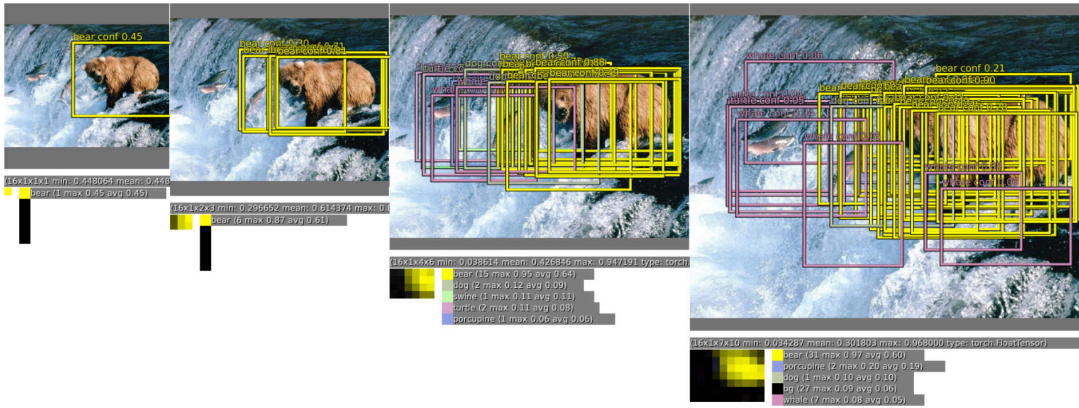
- 分类和回归共享参数



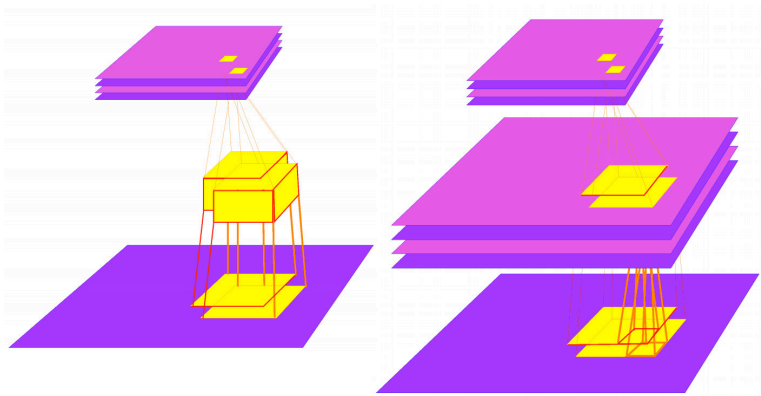
- ILSVRC 2013 定位挑战



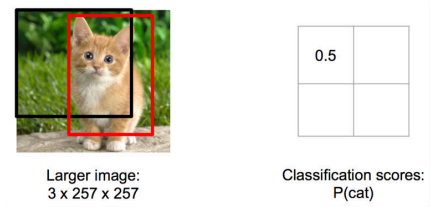
- 基于回归找框


bear conf 0.45
16x1x1x1 min: 0.448064 mean: 0.448
bear (1 max 0.45 avg 0.45)
16x1x2x3 min: 0.296652 mean: 0.614374 max: 0.
bear (6 max 0.87 avg 0.61)
16x1x4x6 min: 0.038614 mean: 0.426848 max: 0.947191 type: torch
bear (15 max 0.95 avg 0.64)
dog (2 max 0.12 avg 0.09)
whale (1 max 0.11 avg 0.11)
turtle (2 max 0.11 avg 0.08)
porcupine (1 max 0.06 avg 0.06)
bear conf 0.21
16x1x7x10 min: 0.034287 mean: 0.301803 max: 0.968000 type: torch.HostTensor)
bear (31 max 0.97 avg 0.60)
porcupine (2 max 0.20 avg 0.19)
dog (1 max 0.10 avg 0.10)
bg (27 max 0.09 avg 0.06)
whale (7 max 0.08 avg 0.05)

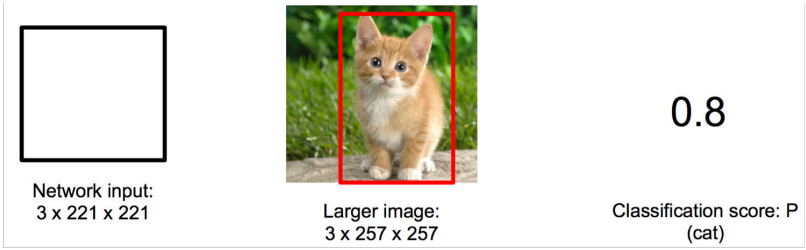
- 滑动窗口 CNN



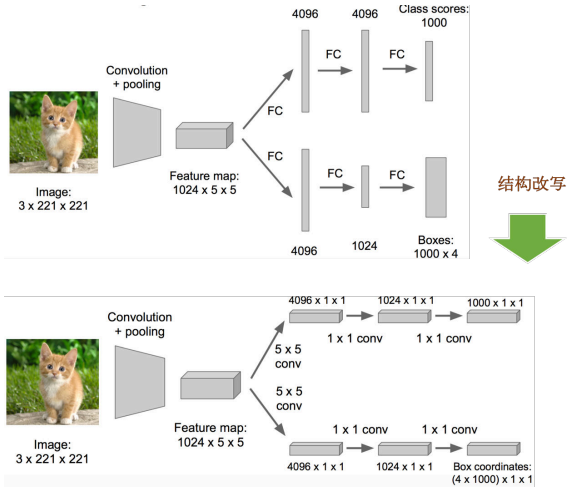
- 滑动窗口
  - 滑动窗口、分类得分、回归定位框



- 合并找到最佳定位框
  - 然后根据合并预测的回归定位框

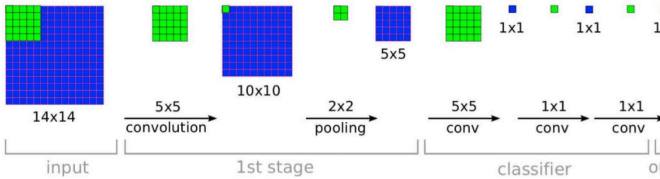


- 修改 FC 网络：高效滑动窗口网络

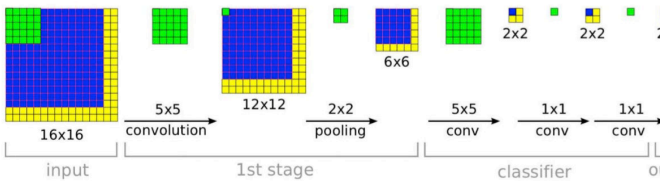


- 高效滑动窗口网络：计算量优势
  - 测试图片的增大额外计算量不大

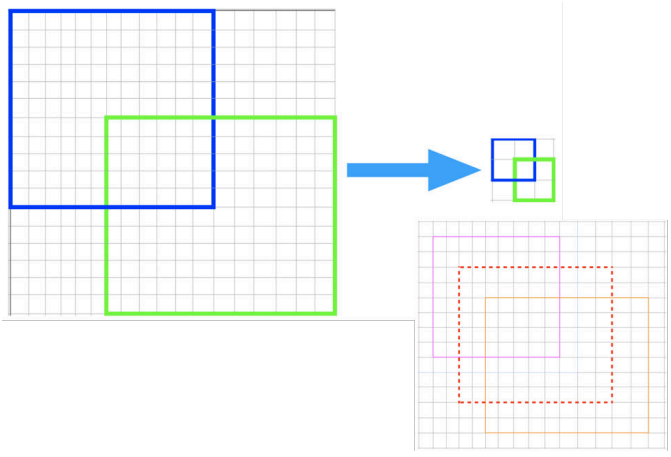
Training time: Small image, 1 x 1 classifier output

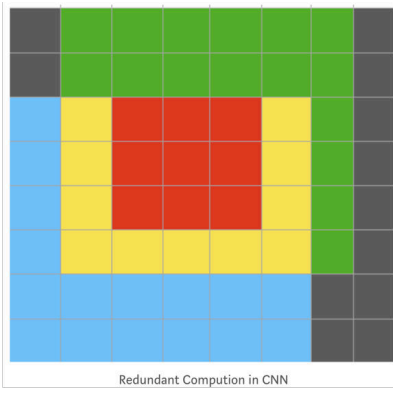


Test time: Larger image, 2 x 2 classifier output, only extra compute at yellow regions

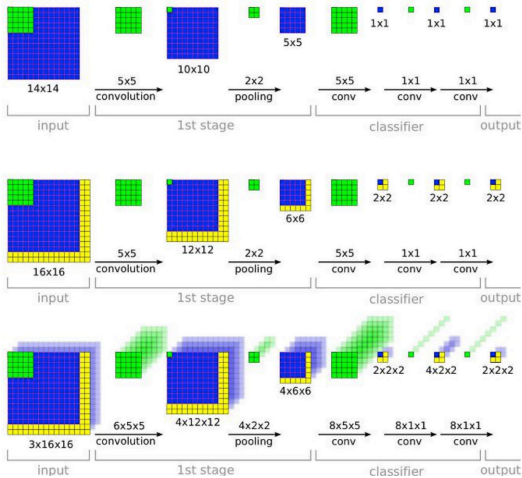


- 重复的特征计算

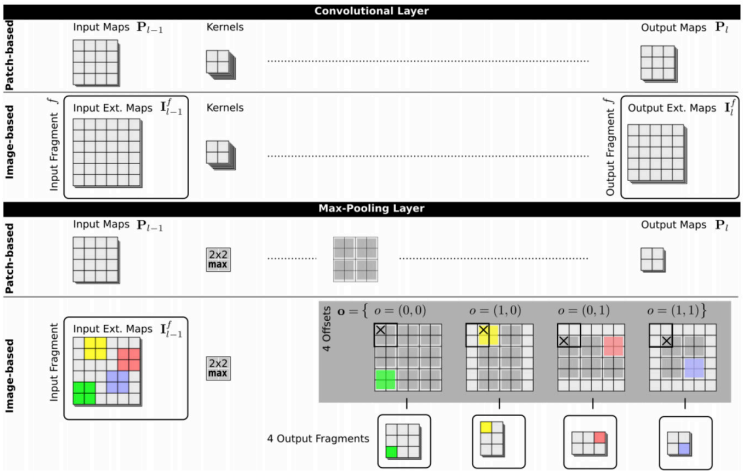




- 多类别定位


14x14
5x5
convolution
10x10
2x2
pooling
5x5
conv
1x1
1x1
conv
1x1
conv
input
1st stage
classifier
output
16x16
5x5
convolution
12x12
2x2
pooling
6x6
5x5
conv
2x2
2x2
2x2
1x1
conv
1x1
conv
input
1st stage
classifier
output
3x16x16
6x5x5
convolution
4x12x12
4x2x2
pooling
4x6x6
8x5x5
conv
2x2x2
4x2x2
2x2x2
8x1x1
conv
8x1x1
conv
input
1st stage
classifier
output

- 思想来源：快速图像扫描
  - 优化处理不同大小的图片



- OverFeat 效果

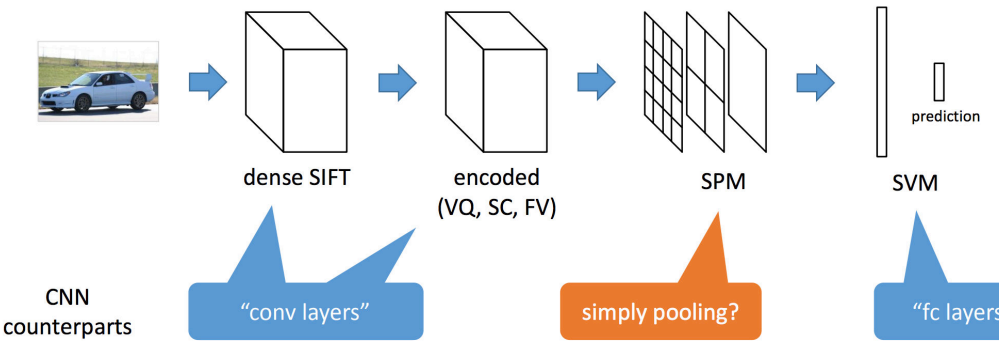

Top predictions:
burrito (confidence 28.9)
Groundtruth:
person
burrito
Top predictions:
burrito (confidence 17.4)
Groundtruth:
burrito

- OverFeat 小结

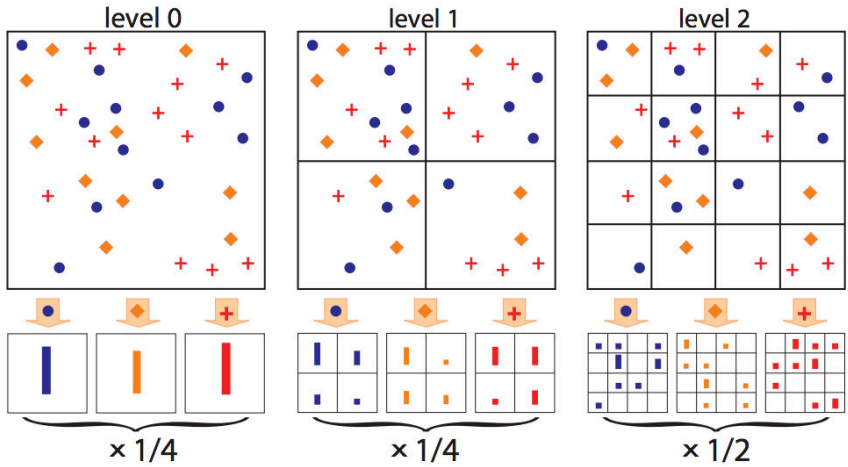
1. 基于回归的定位方式
2. 滑动窗口、缩放图片
3. 定位效果很好、但是对于多个物体分割的情况不佳
4. 另外对于重叠物体，和小的物体的效果不好
5. 对于背景的区分在训练数据上要求比较高
6. 端到端的模型

2.4 SPPNet

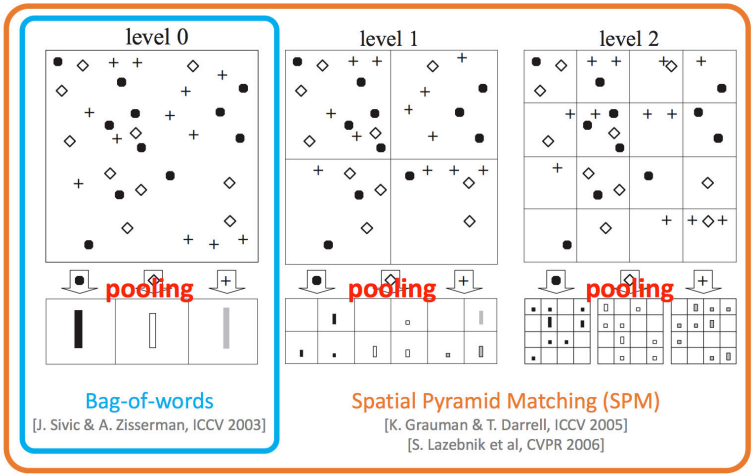
- SPP-net: SPM (Spatial Pyramid Matching) in CNN
  - 传统 SVM 算法到 CNN 算法



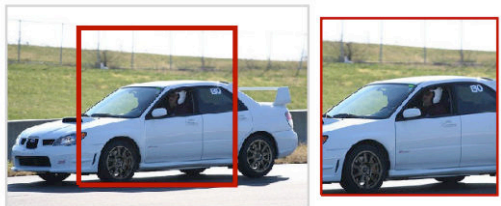
- SPM (Spatial Pyramid Matching)



- 从 BOW(Bag-of-words) 到 SPM (Spatial Pyramid Matching)



- 特征区域


crop
warp

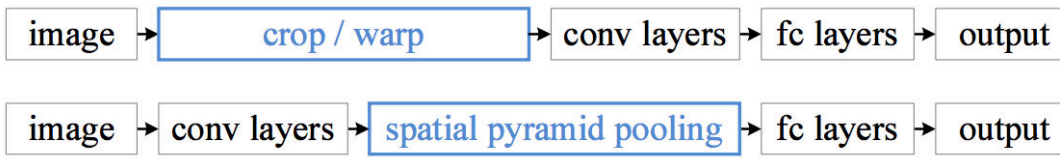


image
crop / warp
conv layers
fc layers
output
image
conv layers
spatial pyramid pooling
fc layers
output

- 特征区域 SPM

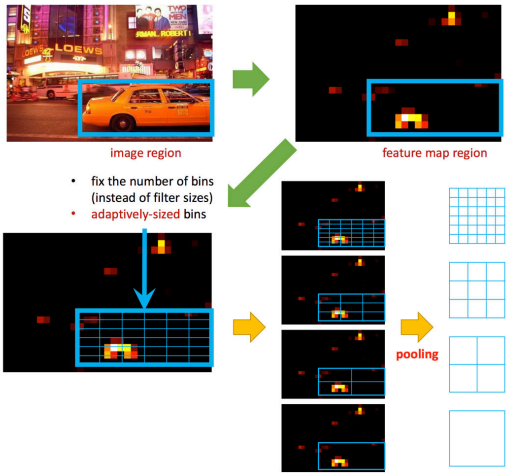
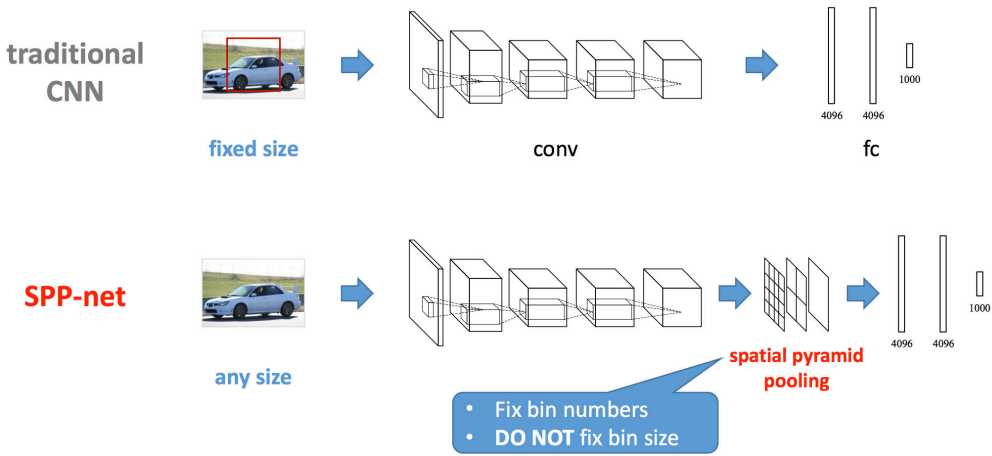
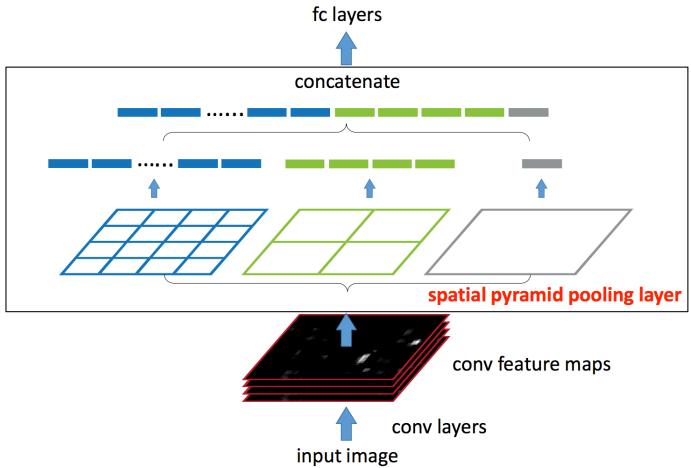


image region
feature map region
fix the number of bins
(instead of filter sizes)
adaptively-sized bins
pooling

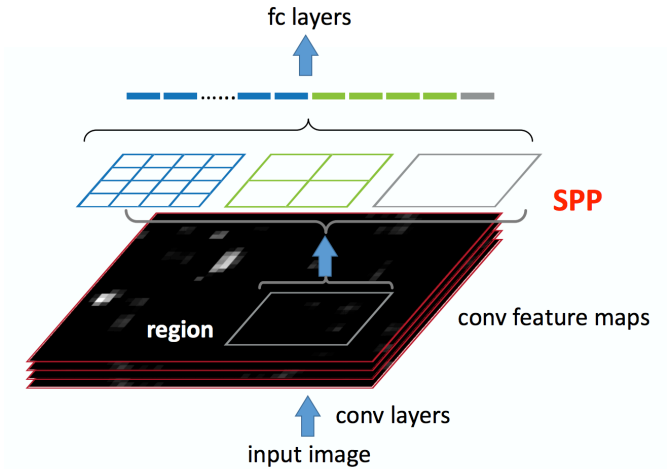
- SPPNet (Spatial Pyramid Pooling Net)



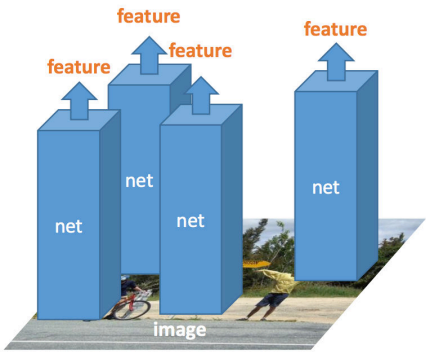
- SPPNet
  - 多层池化


fc layers
concatenate
spatial pyramid pooling layer
conv feature maps
conv layers
input image

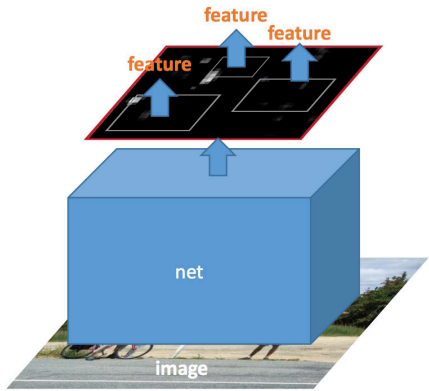
- 区域 SPPNet


fc layers
SPP
region
conv feature maps
conv layers
input image

- 区域 SPPNet 优势
  - 先特征后区域



R-CNN
2000 nets on image regions



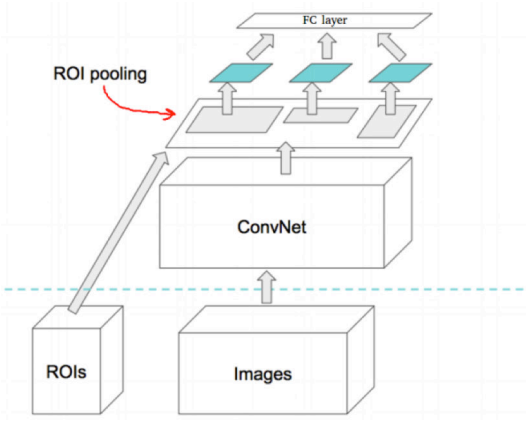
SPP-net
1 net on full image

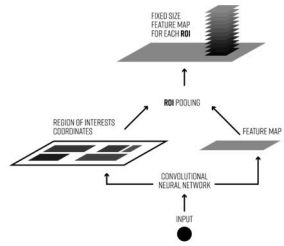
- SPPNet 小结

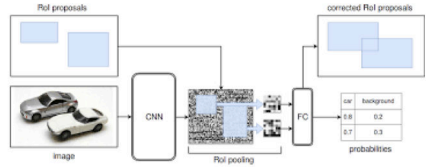
1. 一次输入任意大小图片
2. 任意缩放图片
3. 对于变形比较稳定
4. 建立在 CNN 特征上，速度更快

2.5 Fast R-CNN

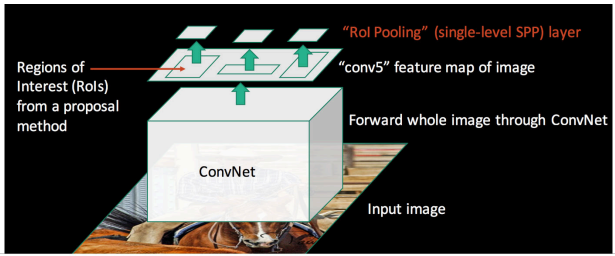
- Region of interest pooling



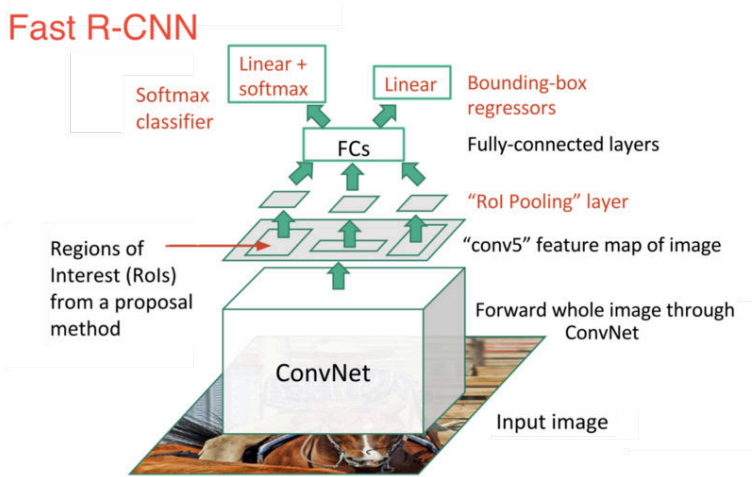




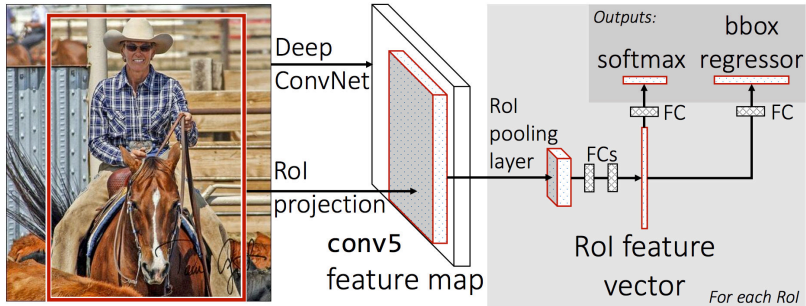
- ROI pooling: 1 层 SPPNet



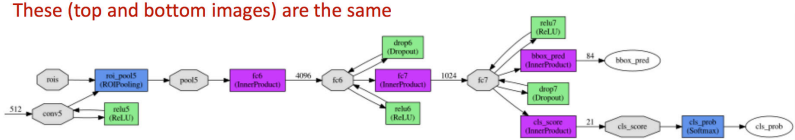
- Fast R-CNN


Fast R-CNN
Linear + softmax
Linear
Softmax classifier
Bounding-box regressors
FCs
Fully-connected layers
"RoI Pooling" layer
Regions of Interest (RoIs) from a proposal method
"conv5" feature map of image
Forward whole image through ConvNet
ConvNet
Input image

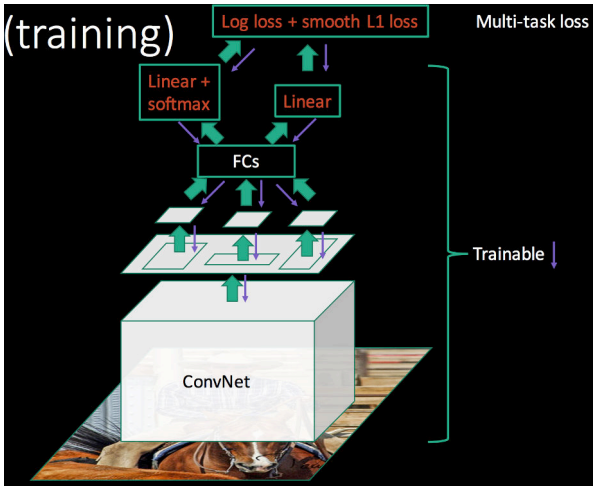
- Fast R-CNN



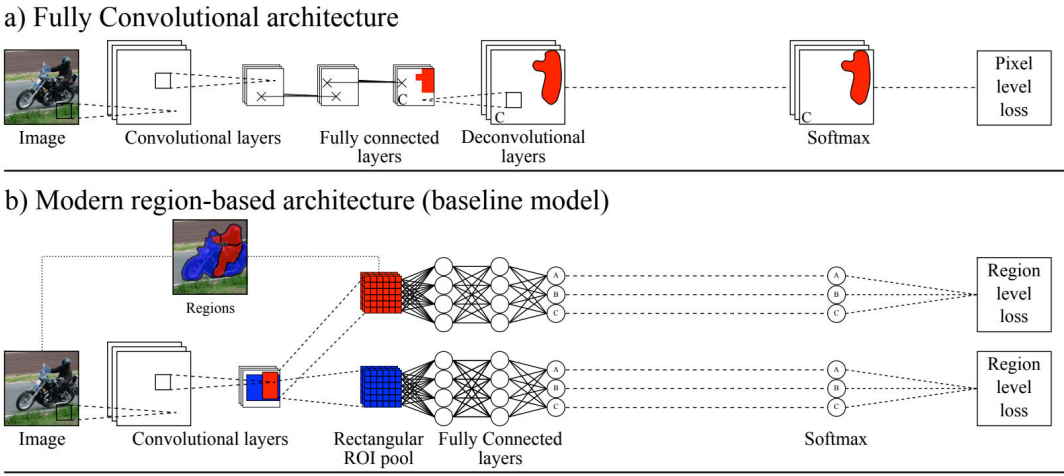
These (top and bottom images) are the same



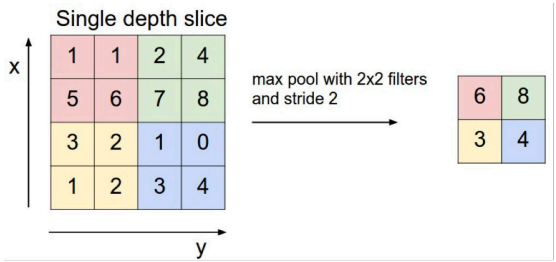
- Fast R-CNN 损失函数：Log 损失 + 光滑的 L1 损失

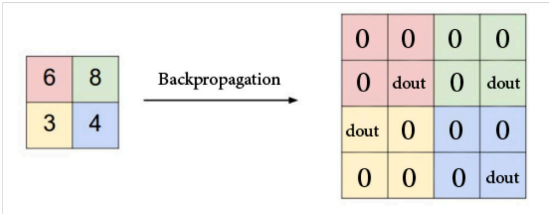


- 基于区域的新架构

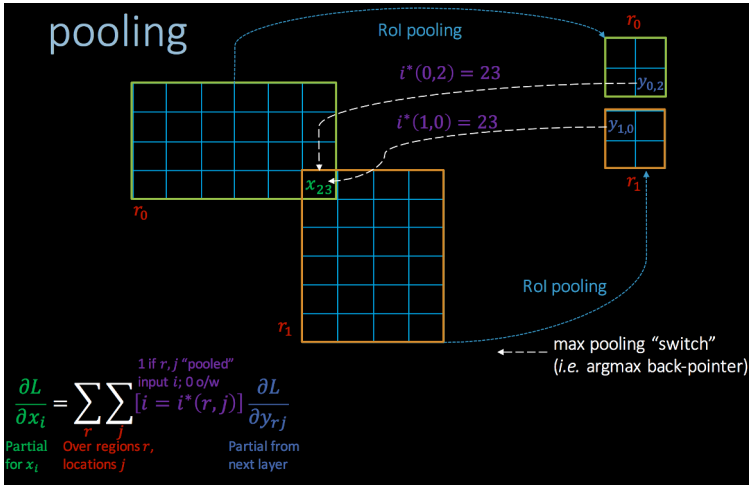

a) Fully Convolutional architecture
Image
Convolutional layers
Fully connected layers
Deconvolutional layers
Softmax
Pixel level loss
b) Modern region-based architecture (baseline model)
Regions
Image
Convolutional layers
Rectangular ROI pool
Fully Connected layers
Softmax
Region level loss
Region level loss

- 池化的反向传播: upsampling 上采样





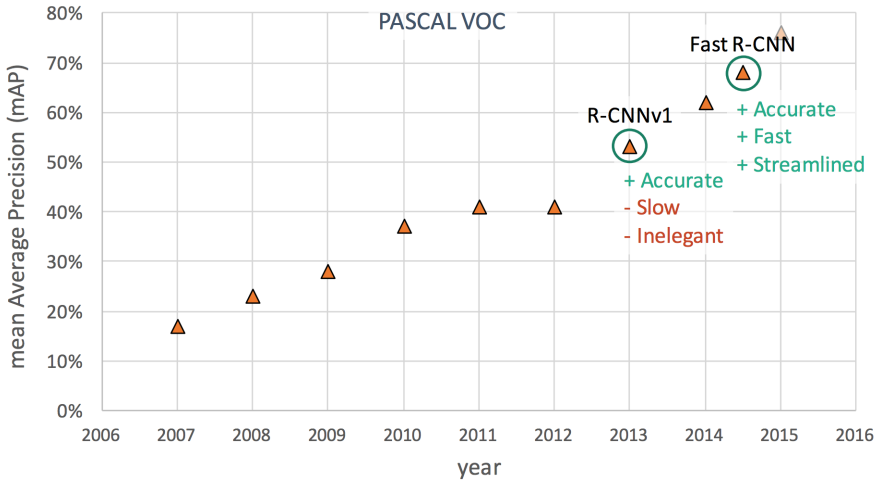
- 反向传播：可求导的池化


pooling
RoI pooling
r_0
$i^*(0,2) = 23$
$y_{0,2}$
$i^*(1,0) = 23$
$y_{1,0}$
x_{23}
r_1
RoI pooling
max pooling "switch"
(*i.e.* argmax back-pointer)
$\frac{\partial L}{\partial x_i} = \sum_r \sum_j [i = i^*(r,j)] \frac{\partial L}{\partial y_{rj}}$
1 if r, j "pooled" input i; 0 o/w
Partial for x_i
Over regions r, locations j
Partial from next layer

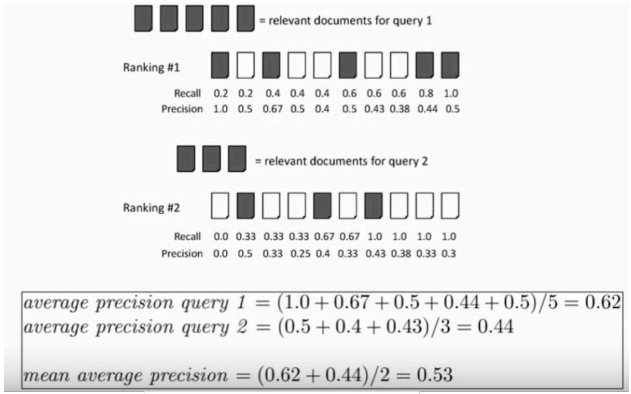
- 速度大幅度提升

		R-CNN	Fast R-CNN
Faster!	Training Time:	84 hours	**9.5 hours**
	(Speedup)	1x	**8.8x**
FASTER!	Test time per image	47 seconds	**0.32 seconds**
	(Speedup)	1x	**146x**
Better!	mAP (VOC 2007)	66.0	**66.9**

- 效果也有提升：整合效应


PASCAL VOC
mean Average Precision (mAP)
80%
70%
60%
50%
40%
30%
20%
10%
0%
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
year
R-CNNv1
+ Accurate
- Slow
- Inelegant
Fast R-CNN
+ Accurate
+ Fast
+ Streamlined

- Mean Average Precision (mAP)



$$\begin{aligned} &\textit{average precision query 1} = (1.0 + 0.67 + 0.5 + 0.44 + 0.5)/5 = 0.62 \\ &\textit{average precision query 2} = (0.5 + 0.4 + 0.43)/3 = 0.44 \\ &\textit{mean average precision} = (0.62 + 0.44)/2 = 0.53 \end{aligned}$$

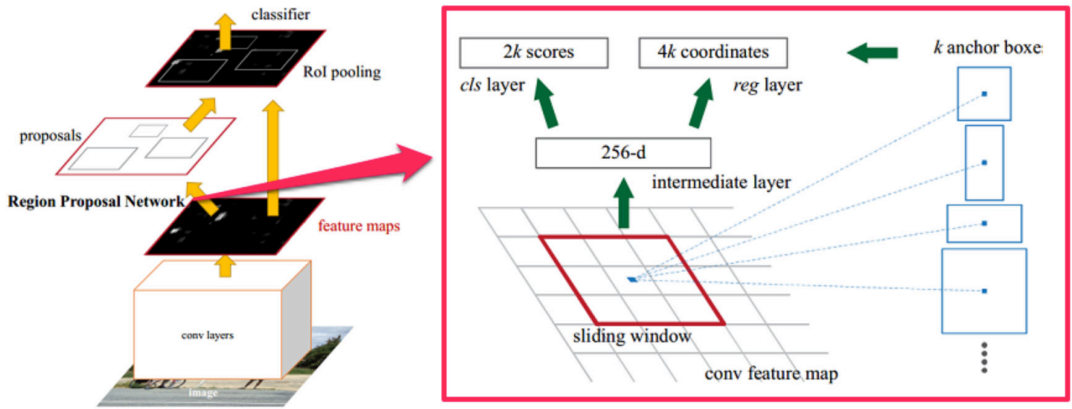
$$\mathbf{MAP} = \frac{\sum_{q=1}^{Q} \mathbf{AveP(q)}}{Q}$$

- Fast R-CNN 小结

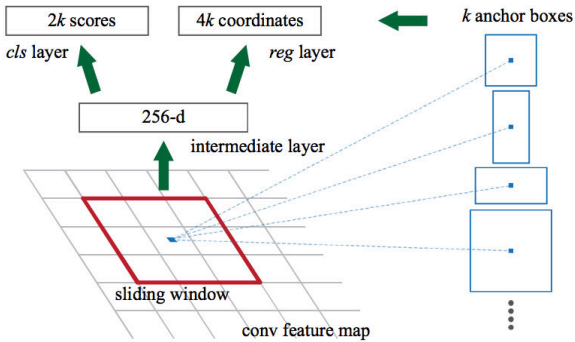
1. SPPNet 和 R-CNN 和集成
2. ROI Pooling 避免大量重复的特征计算
3. 抛弃了 SVM 分类
4. 速度大幅度提升，还带来学习的整合性能提升
5. 但是依然依赖外部的区域推荐算法，不是严格端到端的算法

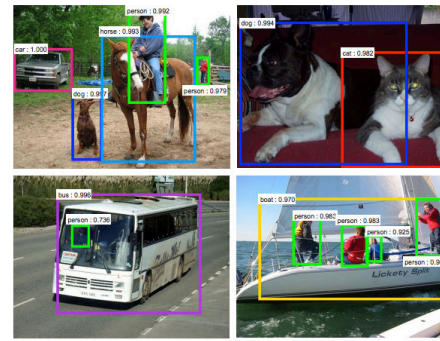
2.6 Faster R-CNN

- RPN(Region Proposal Network) 取代外部区域建议

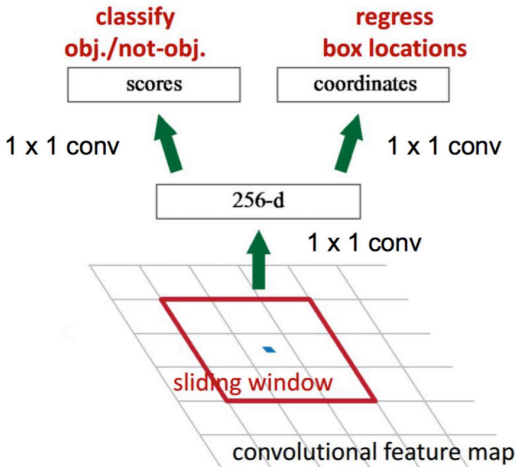


- RPN 基于 CNN 特征

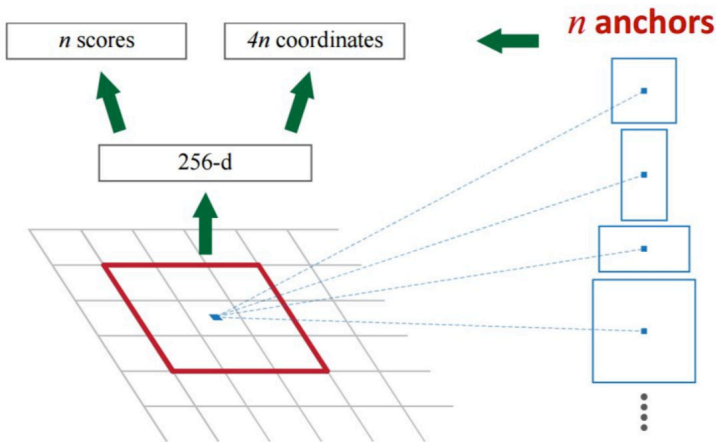

2k scores
4k coordinates
k anchor boxes
cls layer
reg layer
256-d
intermediate layer
sliding window
conv feature map


car : 1.000
person : 0.992
horse : 0.993
dog : 0.97
person : 0.979
dog : 0.994
cat : 0.982
bus : 0.996
person : 0.736
boat : 0.970
person : 0.982
person : 0.983
person : 0.925
person : 0.969

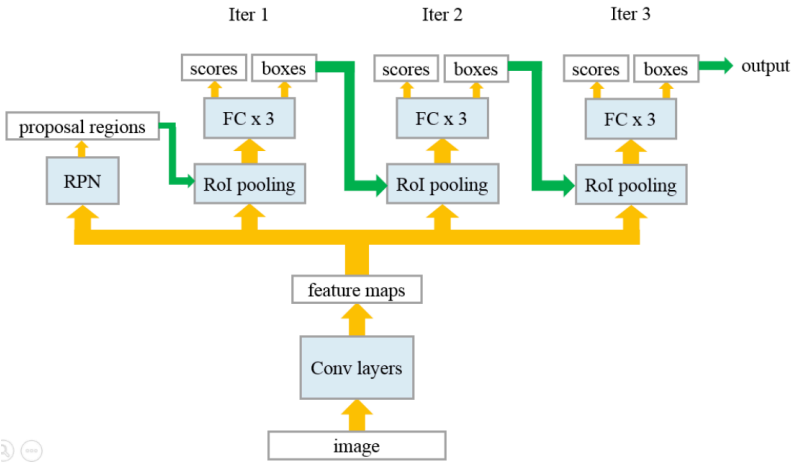
- RPN 实现也是基于分类和回归



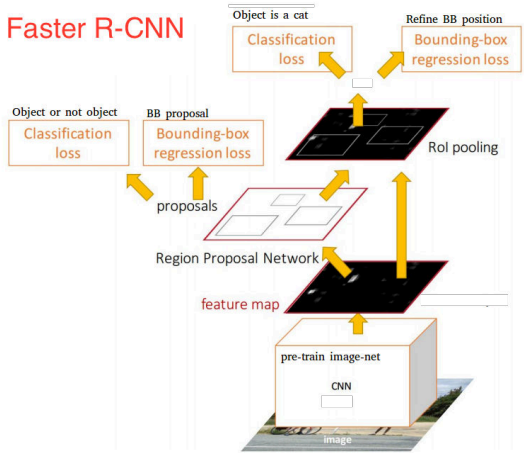
- RPN 每个位置做预设 anchor 个数的尝试



- RPN 输出到 ROI Pooling



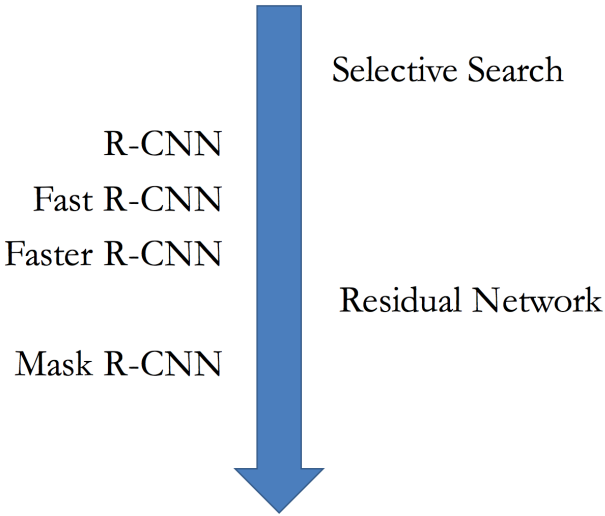
- Faster R-CNN
  - 开始 RPN 的损失和 Fast R-CNN 损失分开计算，后来集成一起计算



- Faster R-CNN 效果

	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
(Speedup)	1x	25x	**250x**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

- R-CNN 系列

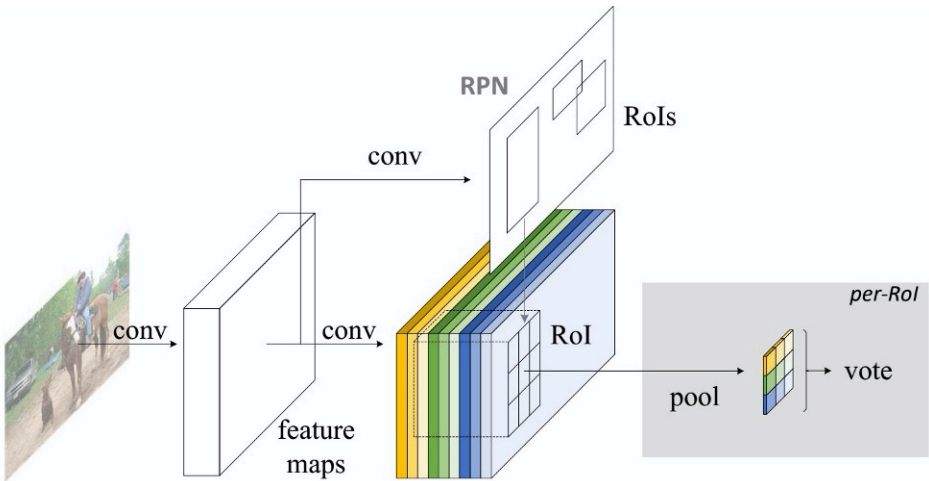

Selective Search
R-CNN
Fast R-CNN
Faster R-CNN
Residual Network
Mask R-CNN

- Faster R-CNN 小结

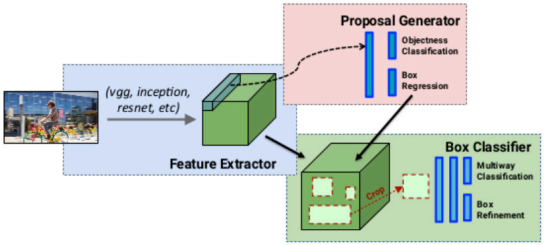
1. RPN 和 Fast R-CNN 和集成
2. 速度更快，效果类似
3. 如何集成事例分割？

2.7 R-FCN

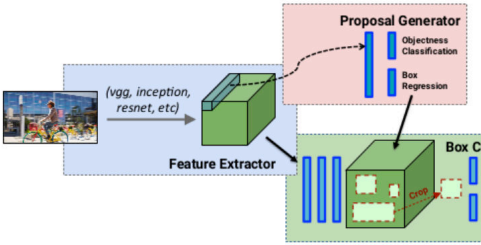
- R-FCN (Region-based Fully Convolutional Networks)



- R-FCN 进一步将特征前置
  - 共享尽可能多的特征计算

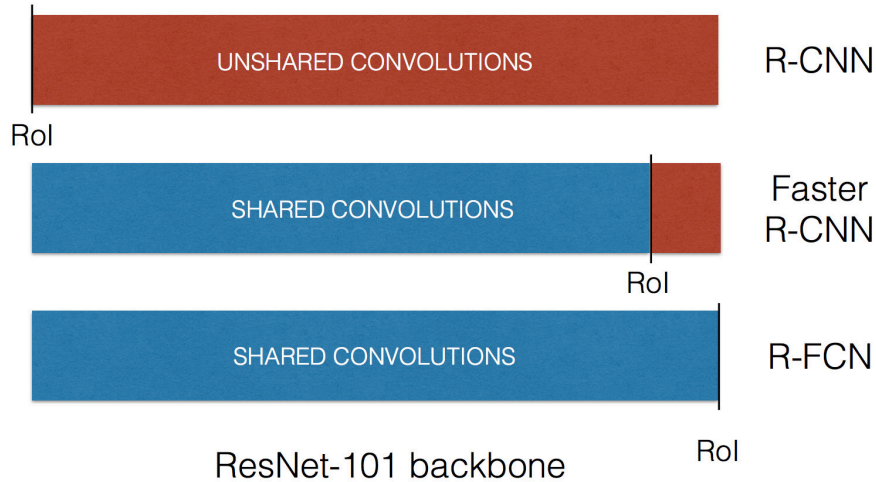


(b) Faster RCNN.

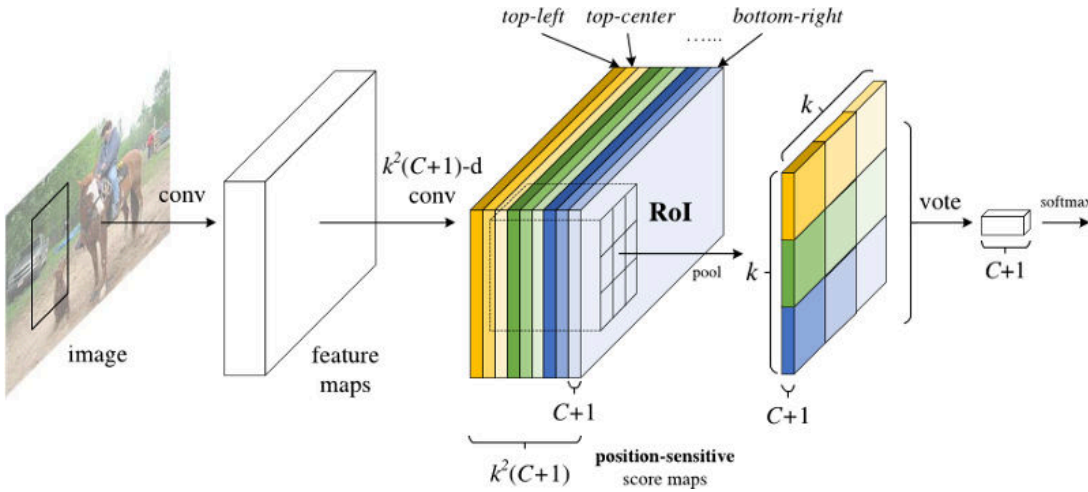


(c) R-FCN.

- R-FCN 的特征计算共享



- R-FCN 提前计算所有需要的特征



- R-FCN 效果
  - 效果类似，但是速度大幅度提升

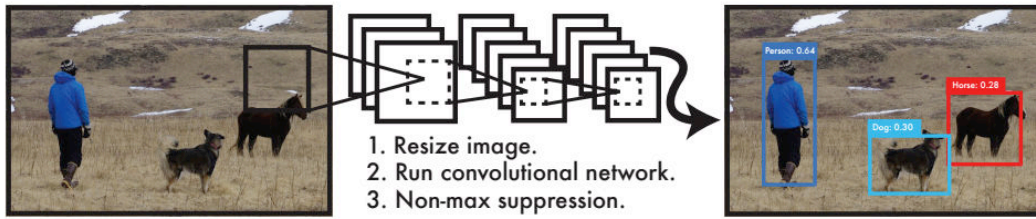
	training data	mAP (%)	test time (sec/img
Faster R-CNN [10]	07++12	73.8	0.42
Faster R-CNN +++ [10]	07++12+COCO	**83.8**	3.36
R-FCN multi-sc train	07++12	77.6†	0.17
R-FCN multi-sc train	07++12+COCO	**82.0**‡	0.17

- R-FCN 小结

1. 基于 Faster R-CNN 进一步提高特征共享
2. 效果和 Faster R-CNN 类似，但是速度更快
3. 进一步专业化特征计算和区域检测的层次

2.8 YOLO

- YOLO (You Only Look Once)



- 一阶段识别

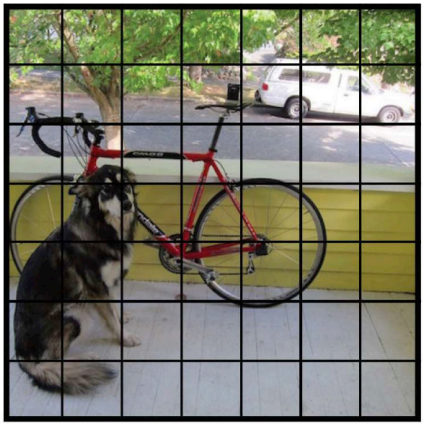
More than one stage

- DetectorNet (Szegedy et al.)
- R-CNN (Girshick et al.)
- SPP-net (He et al.)
- Fast R-CNN (Girshick)
- Faster R-CNN (Ren et al.)
- R-FCN (Dai et al.)
- Mask R-CNN (He et al.)

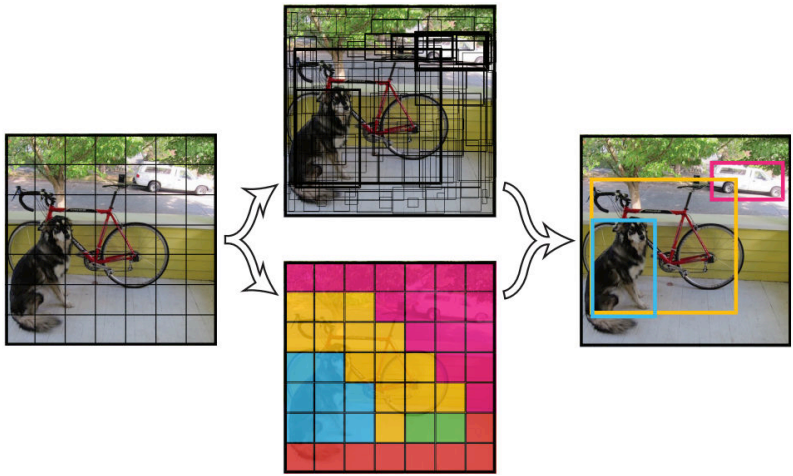
One stage

- OverFeat (Sermanet et al.)
- YOLO, YOLOv2 (Redmon et al.)
- SSD (Wei et al.)
- RetinaNet (Lin et al.) [Poster at WICV on Wed.]

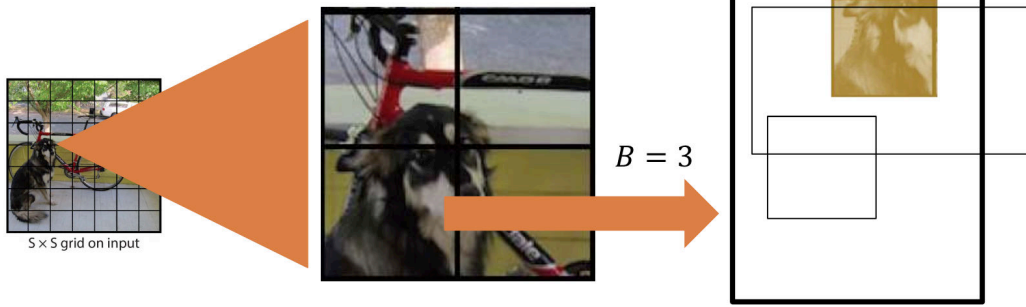
- 分而治之：分块思想



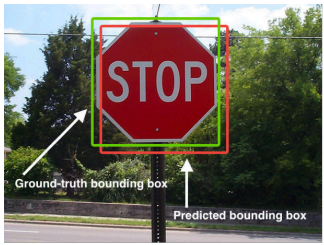
- 分块思想：分块找框 + 分块分类 (类别概率图)



- 分块：回归找框
  - Regression: $x, y, w, h, confidence$
  - $Confidence = \mathrm{P}(Object) * IoU_{pred}^{truth}$

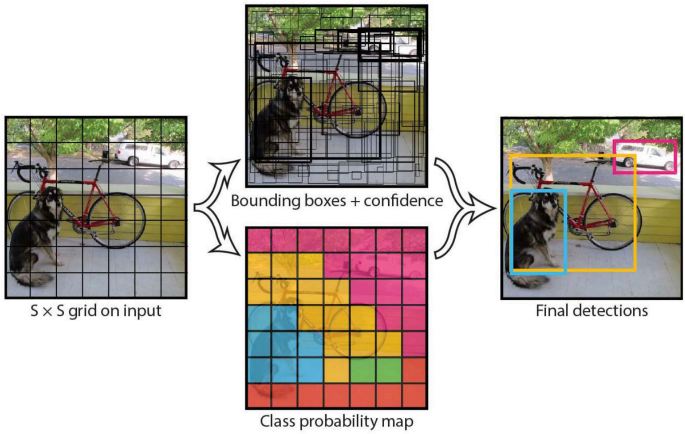


- Intersection over Union (IoU)



$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

- 分块分类结合：回归框的可信度

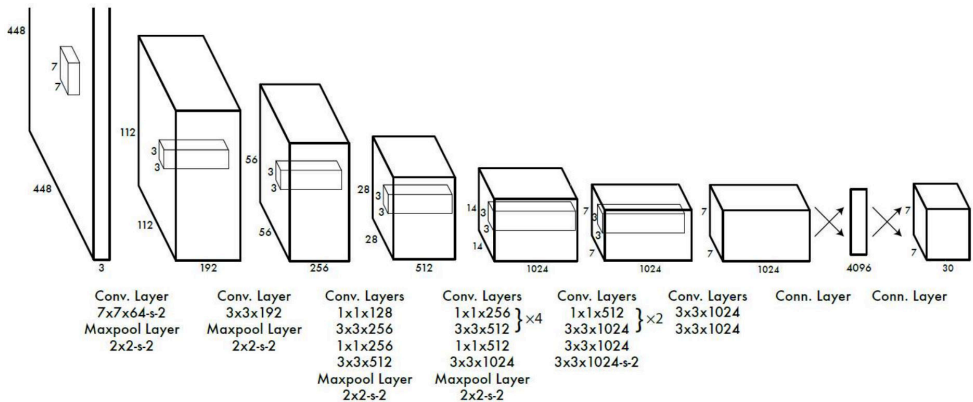


$$\mathrm{P}(Class_i|Object) * \mathrm{P}(Object) * IoU_{pred}^{truth} = \mathrm{P}(class_i) * IoU_{pred}^{truth}$$

- 损失函数：回归框 + 物体分类

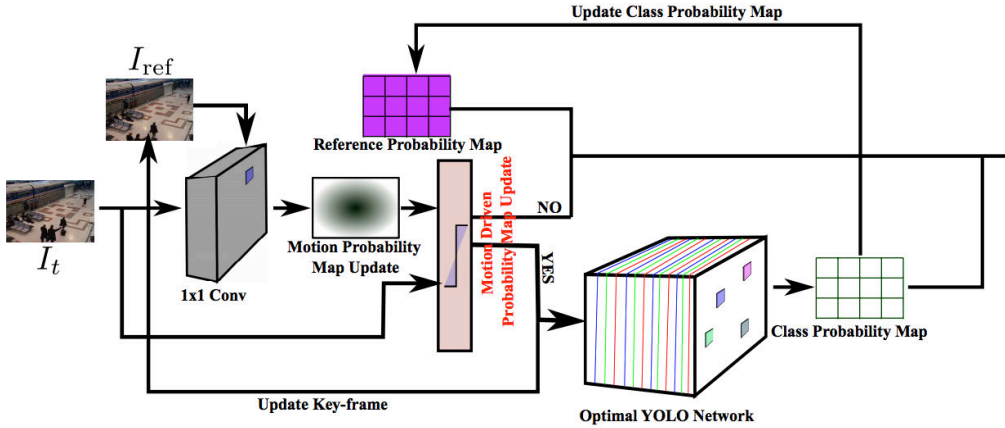
$$
\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(x_i - \hat{x}_i\right)^2 + \left(y_i - \hat{y}_i\right)^2 \\
&+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2 \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i\right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i\right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c)\right)^2
\end{aligned}
$$

- 实现框架



- Output: $S \times S \times (B * 5 + C)$ tensor
  - For PASCAL, $S = 7, B = 2, C = 20$, we have output shape of $7 \times 7 \times 30$

- Fast YOLO
  - 根据视频连续性特点，复用类别概率图



- YOLO 效果：比 Faster R-CNN 差，但是实时的速度
  - Pascal VOC 2007

9 Conv layers

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

- YOLO 小结

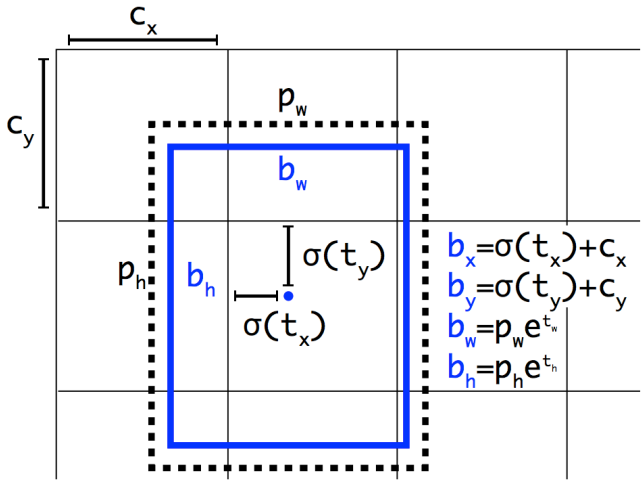
1. 典型的回归加分类模型
2. 分治思想很好
3. 实时性很好，但是准确率不高
4. 单一的 CNN 网络
5. 小物体，不规则物体识别差
6. 定位精度不高

2.9 YOLO2

- YOLO2 (YOLO 9000) 效果：改进较大
  - YOLO 9000 是 YOLO2 模型的一种实现 (混合数据集合)

	YOLO							YOL
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.**

- 回归中心的预测，和框先验：提高定位精度



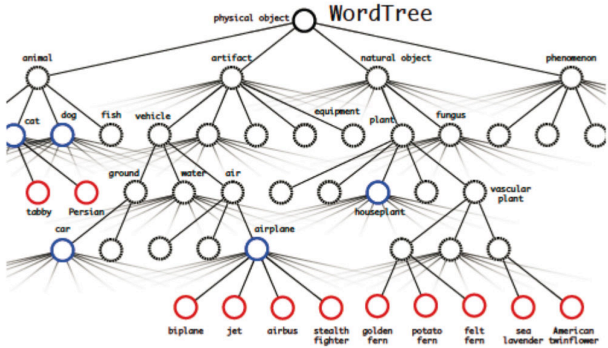
- 新架构 DarkNet19：更快

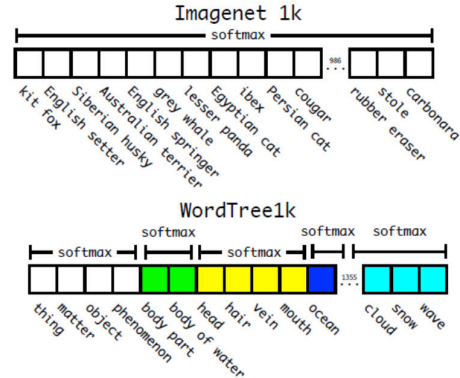
Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	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			

- 技术改进

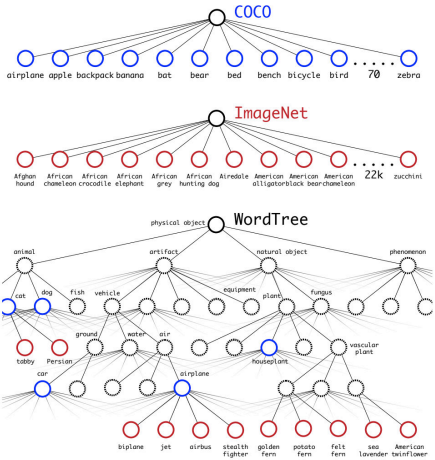
1. 引入 BN(Batch normalization) (%2 mAP 改进)
2. 高分辨率图片 (448x448)，改善小物体识别 (4% mAP 改进)
3. 更细化的分块 (13x13) (1% mAP 改进)
4. 引入 Anchor 框 (K-means) (81% 召回到 88% 召回)
5. 分层的结果标签

- 分层物体类标签:wordtree

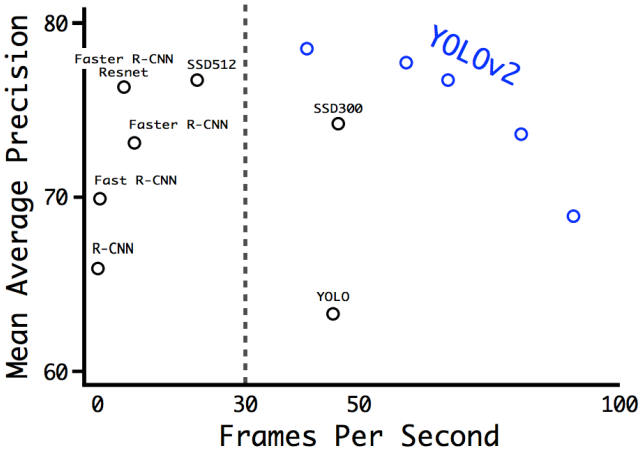




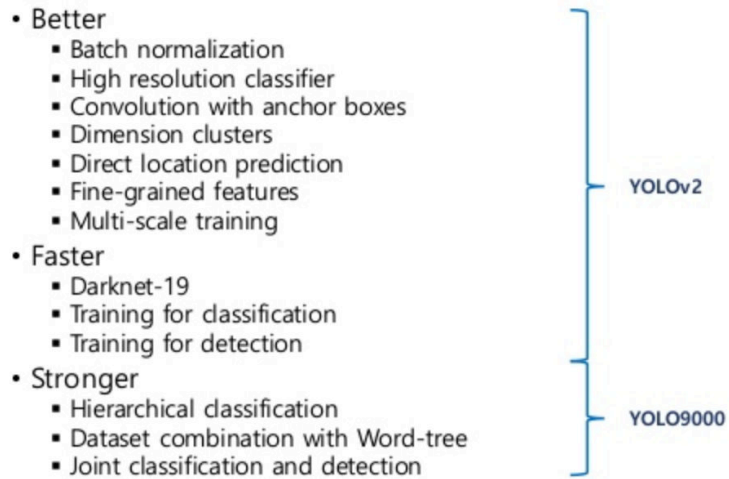
- YOLO9000 的 wordtree：混合多种数据集合



- 可配置的 YOLO2


Mean Average Precision
80
70
60
0
30
50
100
Frames Per Second
Faster R-CNN
Resnet
SSD512
Faster R-CNN
Fast R-CNN
R-CNN
SSD300
YOLO
YOLOv2

- 要点回顾

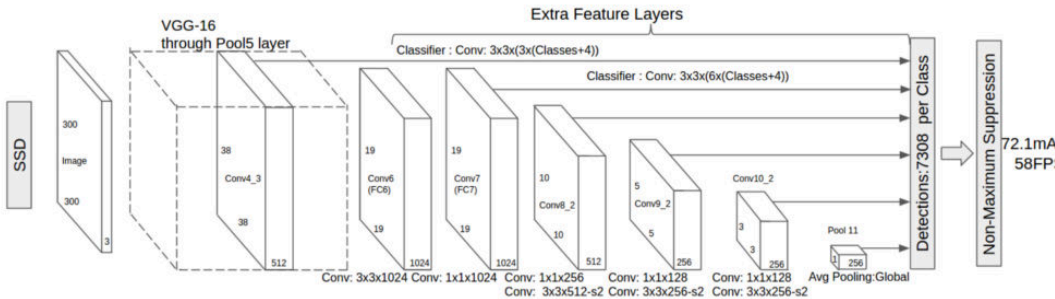

Better
Batch normalization
High resolution classifier
Convolution with anchor boxes
Dimension clusters
Direct location prediction
Fine-grained features
Multi-scale training
Faster
Darknet-19
Training for classification
Training for detection
Stronger
Hierarchical classification
Dataset combination with Word-tree
Joint classification and detection
YOLOv2
YOLO9000

- YOLO2 小结

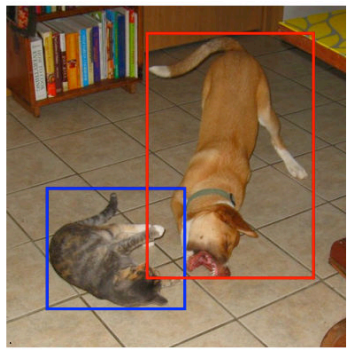
1. 大量技术改进，允许效果和速度的调节
2. 引入 Anchor，精度提高
3. 引入 Wordtree, 类别标签细化
4. 单阶段快速实时识别的代表
5. 依然难以达到即快又准

2.10 SSD

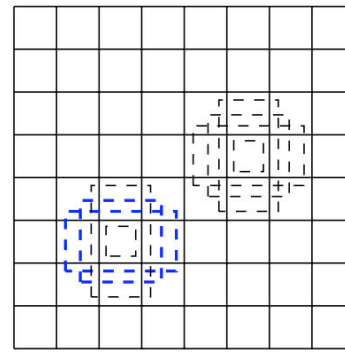
- SSD (Single Shot Detector)



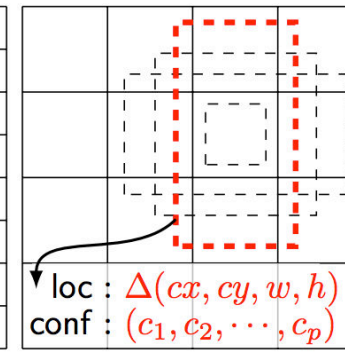
- SSD: 多尺度特征映射 + 多默认比例框



(a) Image with GT boxes

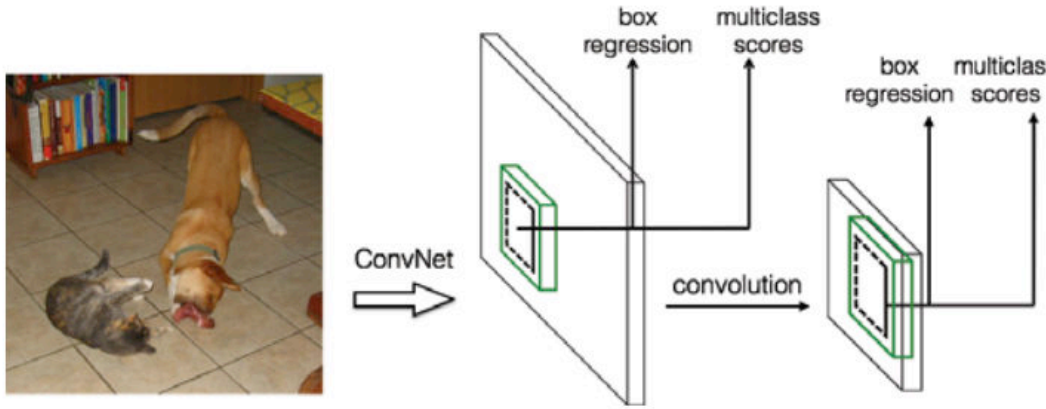


(b) 8×8 feature map

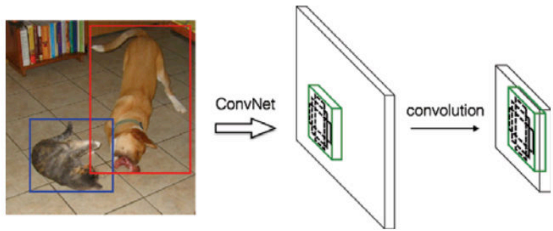


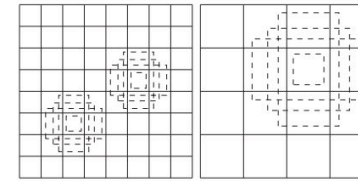
(c) 4×4 feature map

- SSD: 多层 CNN 网络实现多尺度特征映射

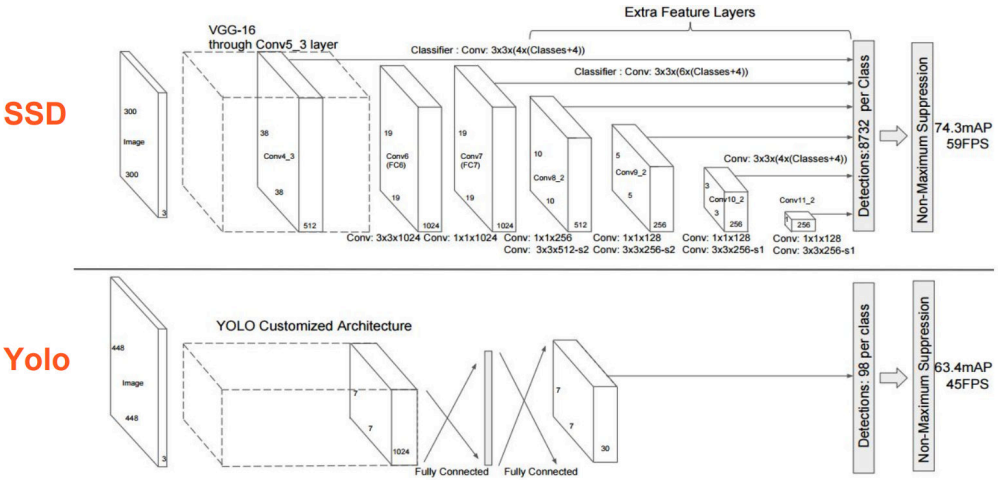


- SSD: 更多默认比例的框 (Anchor Box)





- SSD vs YOLO


SSD
VGG-16
through Conv5_3 layer
Extra Feature Layers
Classifier : Conv: 3x3x(4x(Classes+4))
Classifier : Conv: 3x3x(6x(Classes+4))
Conv: 3x3x(4x(Classes+4))
Image
Conv4_3
Conv6 (FC6)
Conv7 (FC7)
Conv8_2
Conv9_2
Conv10_2
Conv11_2
Conv: 3x3x1024
Conv: 1x1x1024
Conv: 1x1x256
Conv: 3x3x512-s2
Conv: 1x1x128
Conv: 3x3x256-s2
Conv: 1x1x128
Conv: 3x3x256-s1
Conv: 1x1x128
Conv: 3x3x256-s1
Detections:8732 per Class
Non-Maximum Suppression
74.3mAP
59FPS
Yolo
YOLO Customized Architecture
Image
Fully Connected
Fully Connected
Detections: 98 per class
Non-Maximum Suppression
63.4mAP
45FPS

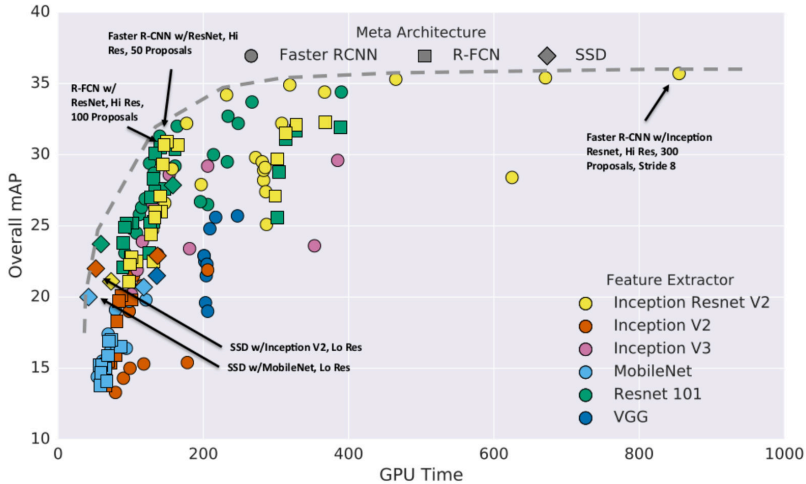
- SSD 效果：比 YOLO 更快更好

	SSD*	Yolo*	Faster R-CNN*
mAP	74.3	66.4	73.2
FPS	46	21	7

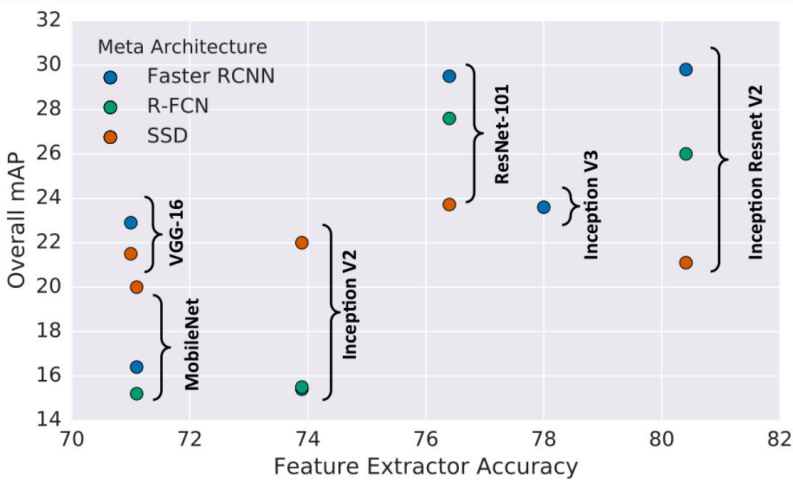
*VGG16

Trained on Pascal VOC 2007 + 2012 dataset

- SSD 系列和 R-CNN 系列对比



- CNN 网络结构的影响

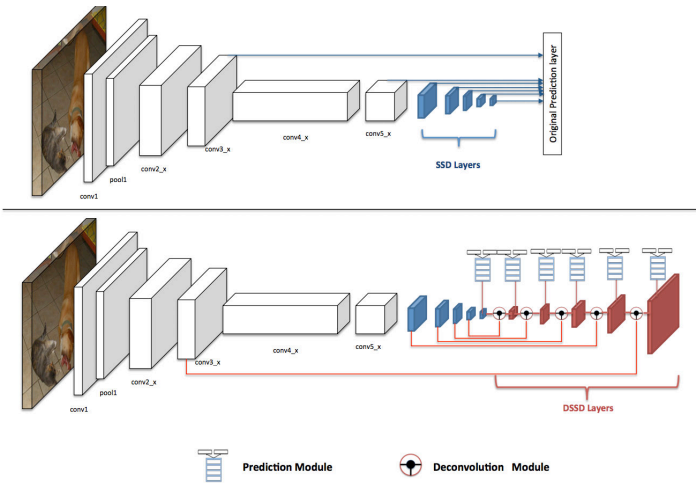

Meta Architecture
Faster RCNN
R-FCN
SSD
VGG-16
MobileNet
Inception V2
ResNet-101
Inception V3
Inception Resnet V2
Overall mAP
Feature Extractor Accuracy
32
30
28
26
24
22
20
18
16
14
70
72
74
76
78
80
82

- SSD 小结

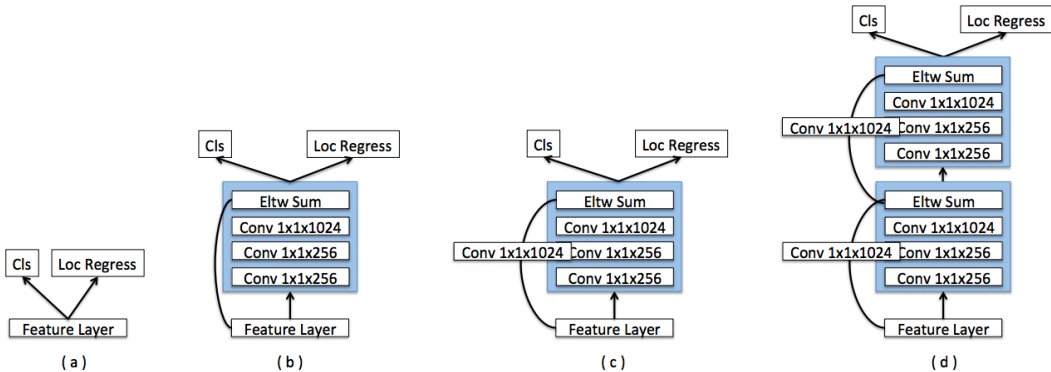
1. 在 YOLO 基础上引入多尺度特征映射
2. 引入 Anchor Box 机制
3. 效果更好，速度更快

2.11 DSSD

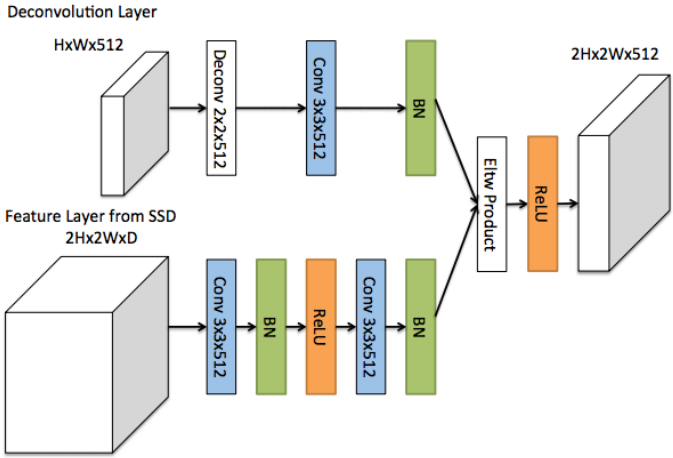
- DSSD (Deconvolutional Single Shot Detector)



- 残差性质预测网络


Cls
Loc Regress
Feature Layer
(a)
Cls
Loc Regress
Eltw Sum
Conv 1x1x1024
Conv 1x1x256
Conv 1x1x256
Feature Layer
(b)
Cls
Loc Regress
Eltw Sum
Conv 1x1x1024
Conv 1x1x1024
Conv 1x1x256
Conv 1x1x256
Feature Layer
(c)
Cls
Loc Regress
Eltw Sum
Conv 1x1x1024
Conv 1x1x1024
Conv 1x1x256
Conv 1x1x256
Eltw Sum
Conv 1x1x1024
Conv 1x1x1024
Conv 1x1x256
Conv 1x1x256
Feature Layer
(d)

- DeConv 层模块


Deconvolution Layer
HxWx512
Deconv 2x2x512
Conv 3x3x512
BN
2Hx2Wx512
Eltw Product
ReLU
Feature Layer from SSD
2Hx2WxD
Conv 3x3x512
BN
ReLU
Conv 3x3x512
BN

- DSSD 效果

Method	network	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow
Faster [24]	VGG	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9
ION [1]	VGG	75.6	79.2	83.1	77.6	65.6	54.9	85.4	85.1	87.0	54.4	80.6
Faster [14]	Residual-101	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	**89.8**	56.7	87.8
MR-CNN [10]	VGG	78.2	80.3	84.1	78.5	70.8	68.5	88.0	85.9	87.8	60.3	85.2
R-FCN [3]	Residual-101	80.5	79.9	**87.2**	81.5	72.0	**69.8**	86.8	88.5	89.8	**67.0**	**88.1**
SSD300*[18]	VGG	77.5	79.5	83.9	76.0	69.6	50.5	87.0	85.7	88.1	60.3	81.5
SSD 321	Residual-101	77.1	76.3	84.6	79.3	64.6	47.2	85.4	84.0	88.8	60.1	82.6
DSSD 321	Residual-101	78.6	81.9	84.9	80.5	68.4	53.9	85.6	86.2	88.9	61.1	83.5
SSD512*[18]	VGG	79.5	84.8	85.1	81.5	73.0	57.8	87.8	88.3	87.4	63.5	85.4
SSD 513	Residual-101	80.6	84.3	87.6	**82.6**	71.6	59.0	88.2	88.1	89.3	64.4	85.6
DSSD 513	Residual-101	**81.5**	**86.6**	86.2	**82.6**	**74.9**	62.5	**89.0**	**88.7**	88.8	65.2	87.0

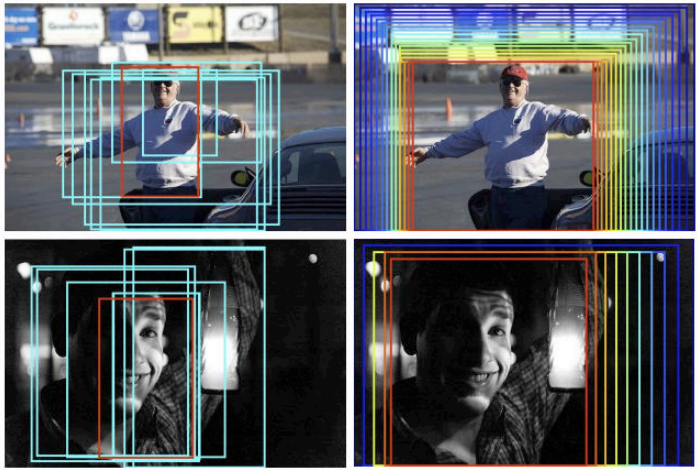
- DSSD 小结

1. 加入 DeConv 模型作为上下文
2. 更深的预测模块
3. 效果提升不大，但是速度变慢

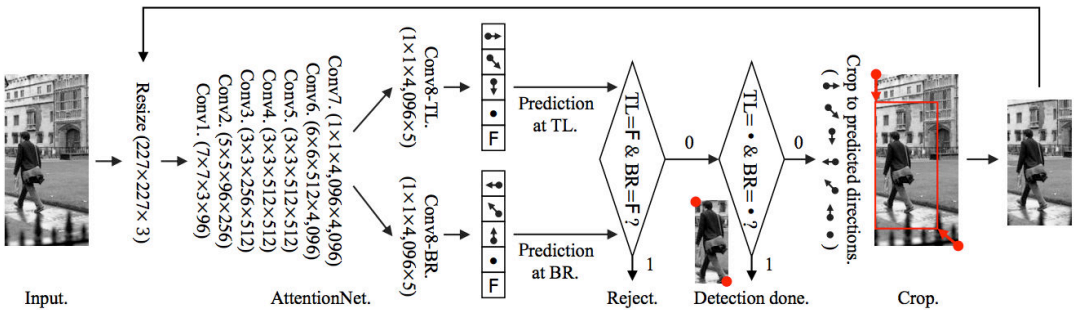
2.12 AttentionNet

- AttentionNet 思想

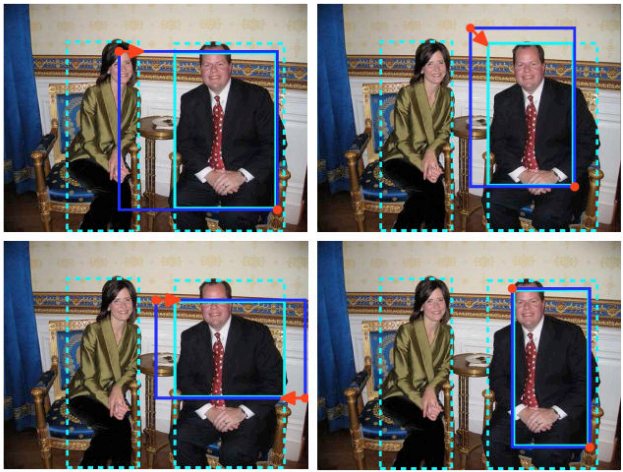
- AttentionNet 和区域推荐对比



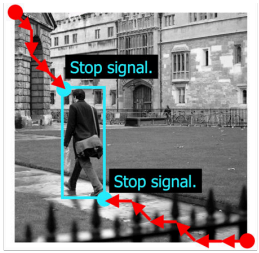
- AttentionNet 架构

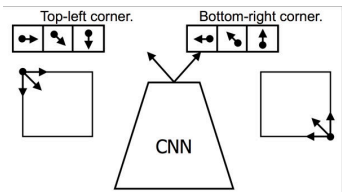


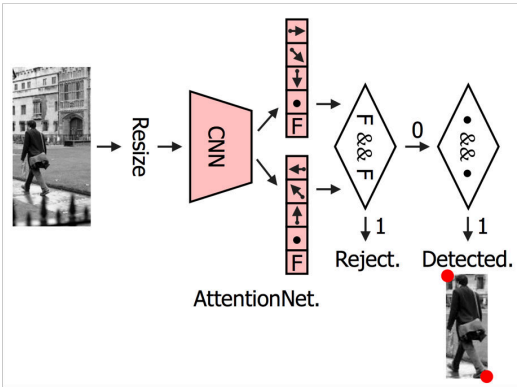
- AttentionNet 注意力移动



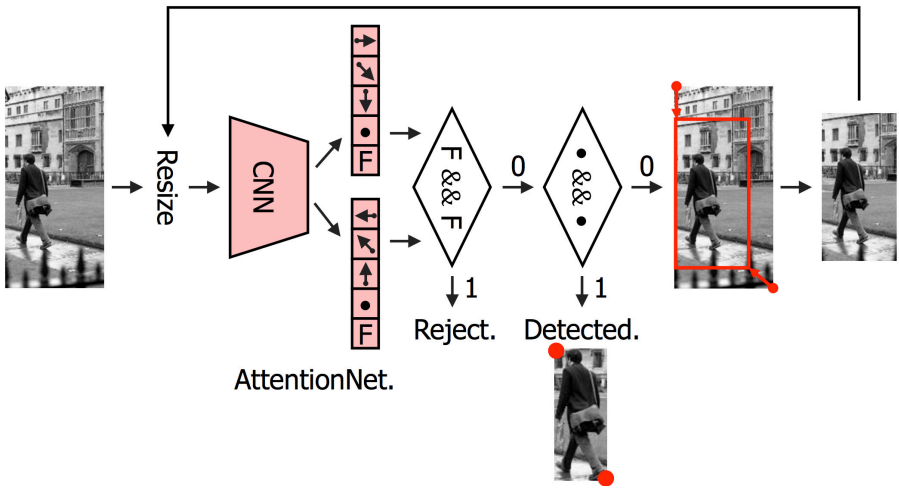
- AttentionNet 基于 CNN 分类的注意力移动



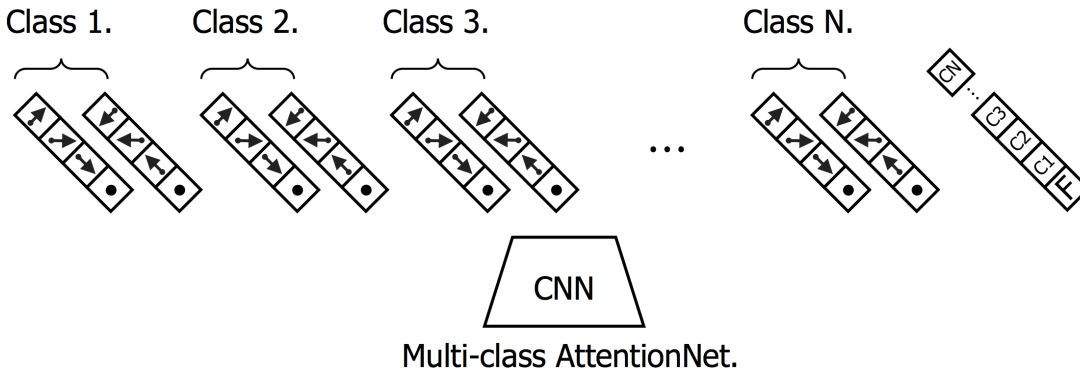




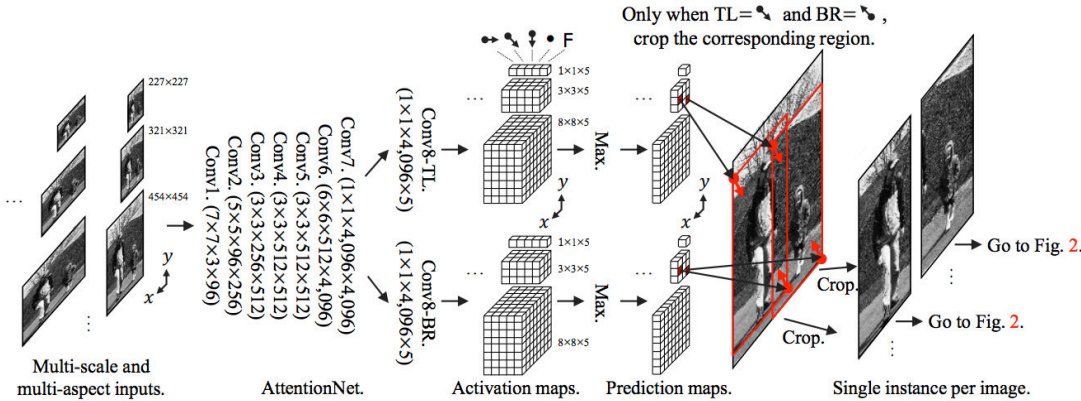
- AttentionNet 循环迭代



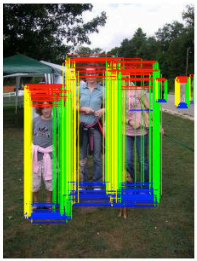
- AttentionNet 多分类



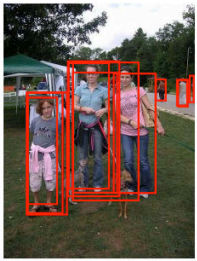
- AttentionNet 多尺度、多方向、裁剪



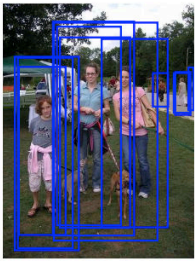
- AttentionNet 多检测框合并



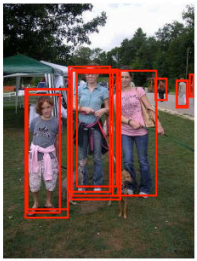
(a) Initial detections.



(b) Initial merge.



(c) Re-initialize. (×2.5)

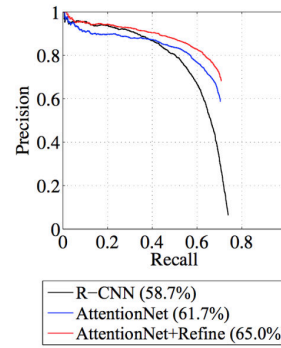


(d) Re-detections.

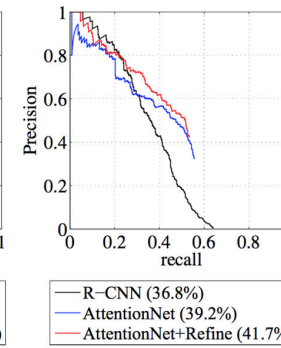
(e) Final merge.

- AttentionNet 效果

Method	Extra data	VOC'07	VOC'12
AttentionNet	ImNet	61.7	62.8
AttentionNet + Refine	ImNet	**65.0**	**65.6**
AttentionNet + R-CNN	ImNet	66.4	69.0
AttentionNet + Refine + R-CNN	ImNet	**69.8**	**72.0**
Person R-CNN + BBReg	ImNet	59.7	N/A
Person R-CNN + BBReg×2	ImNet	**59.8**	N/A
Person R-CNN + BBReg×3	ImNet	59.7	N/A
Felzenszwalb *et al.*'10 [11]	None.	41.9	N/A
Bourdev *et al.*'10 [2]	H3D	46.9	N/A
Szegedy *et al.*'13 [24]	VOC'12	26.2	N/A
Erhan *et al.*'14 [9]	None.	37.5	N/A
Gkioxari *et al.*'14 [13]	VOC'12	45.6	N/A
Bourdev *et al.*'14 [3]	ImNet + H3D	59.3	58.7
He *et al.*'14 [14]	ImNet	57.6	N/A
Girshick *et al.*'14 [12]	ImNet	58.7	57.8
Girshick *et al.*'14 [12]	ImNet	**64.2***	N/A
Shen and Xue '14 [20]	ImNet	59.1	**60.2**



(a) Person class



(b) Bottle class

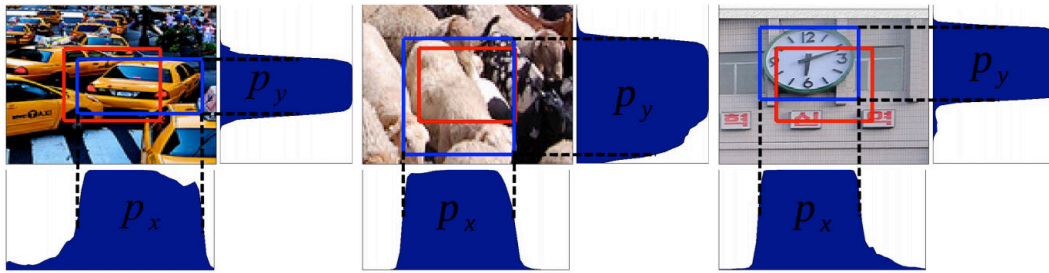
- AttentionNet 小结

1. 全新的区域查找方式
2. 多实例的方式较为复杂
3. 对比 R-CNN，效果有提升

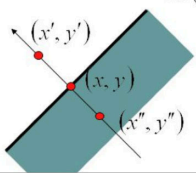
2.13 AttractioNet

- AttractioNet: (Act)ive Box Proposal Generation via (I)n-(O)ut Localization (Net)work
  - 框优化思想

- AttractioNet 框优化

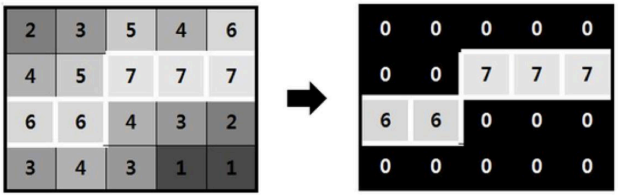


- NMS (non-maxima suppression)

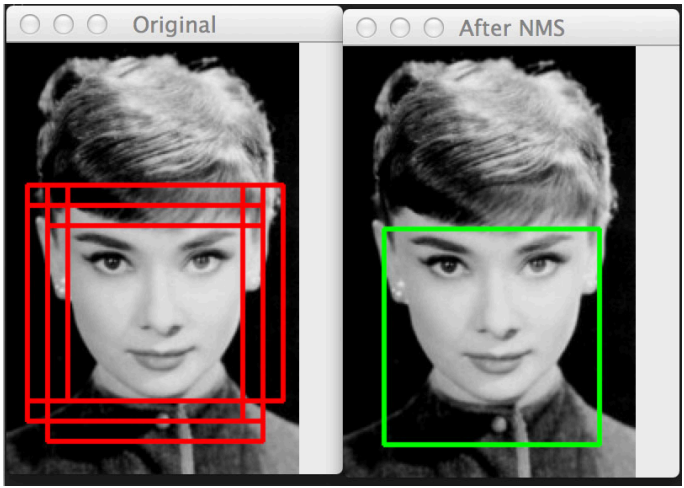


$$M(x,y)=\begin{cases}|\nabla S|(x,y) & \text{if } |\nabla S|(x,y)>|\nabla S|(x',y') \\ & \&\ |\nabla S|(x,y)>|\nabla S|(x'',y'') \\ 0 & \text{otherwise}\end{cases}$$

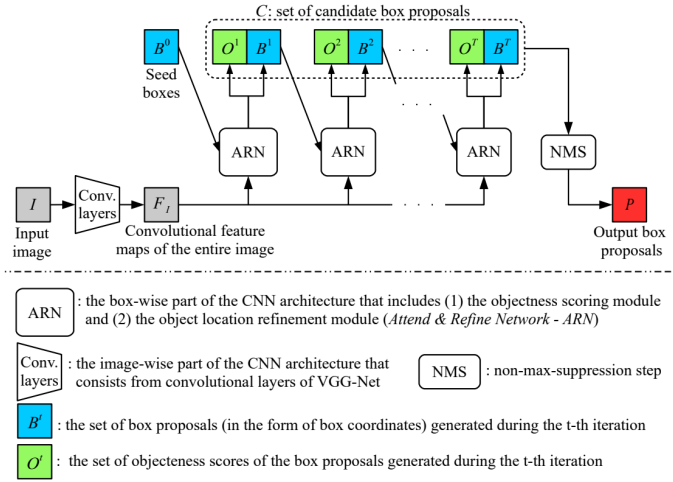
(x',y') and (x'',y'') are the neighbors of (x,y) in $|\nabla S|$ along the direction normal to an edge



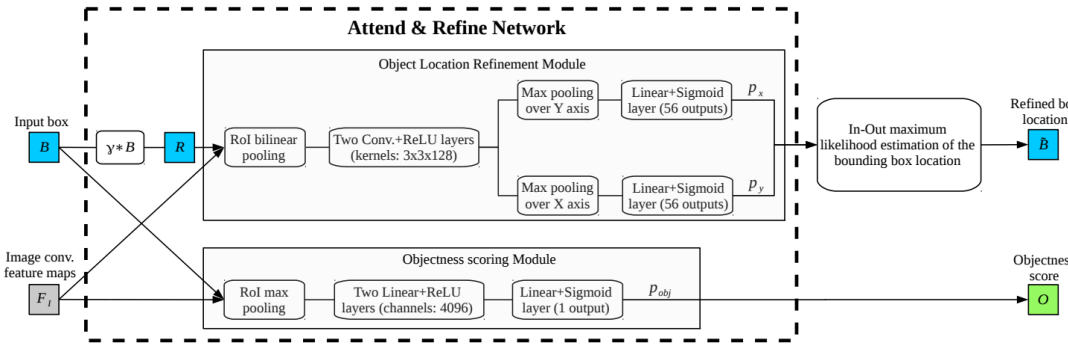
- NMS 效果



- AttractioNet 流程



- Attend & Refine 参与优化



- AttractioNet 效果

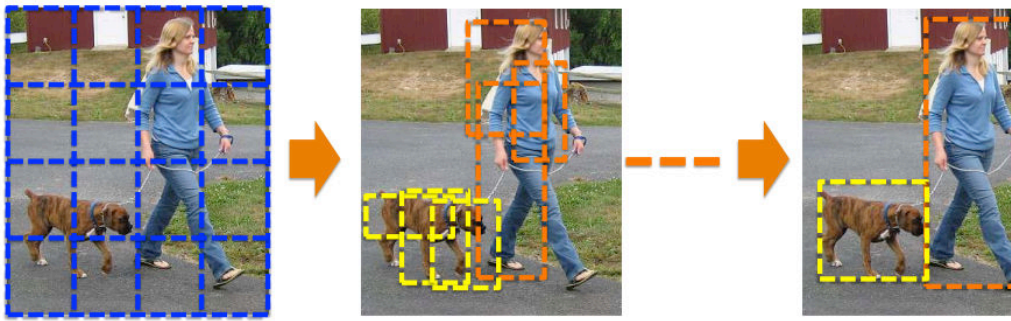
Method	AR@10	AR@100	AR@1000	AR@100-Small	AR@100-Medium	AR@100-Large
AttractioNet (Ours)	**0.159**	**0.389**	**0.579**	**0.205**	**0.419**	**0.498**
EdgeBoxes [42]	0.049	0.160	0.362	0.020	0.131	0.332
Selective Search [40]	0.024	0.143	0.422	0.008	0.085	0.362
MCG [3]	0.078	0.237	0.441	0.045	0.195	0.476

- AttractioNet 小结

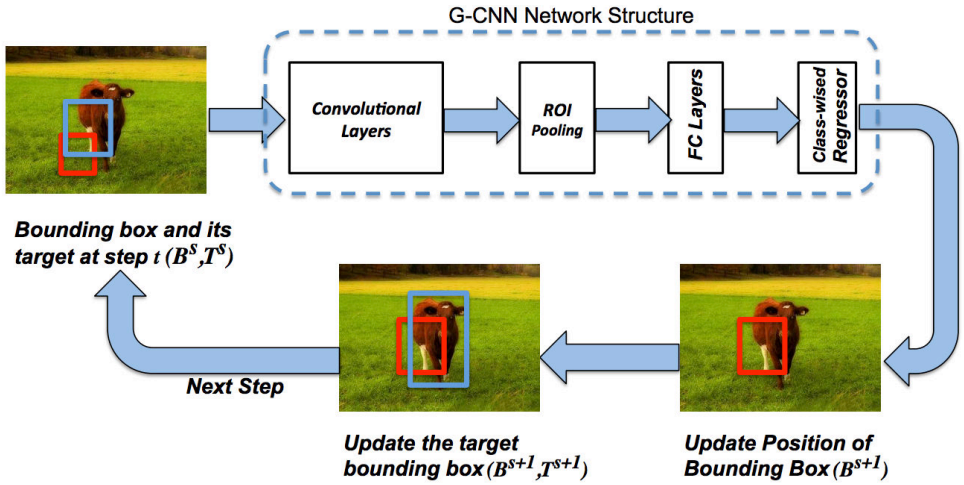
1. 提出迭代优化区域思想
2. AttractioNet 要比 Selective Search 效果更好
3. CNN 网络上的区域推荐

2.14 G-CNN

- G-CNN (Grid CNN)
  - 多尺度回归分块 (网格) + 迭代框优化



- G-CNN 迭代框优化


G-CNN Network Structure
Convolutional Layers
ROI Pooling
FC Layers
Class-wised Regressor
Bounding box and its target at step t (B^s,T^s)
Next Step
Update the target bounding box (B^{s+1},T^{s+1})
Update Position of Bounding Box (B^{s+1})

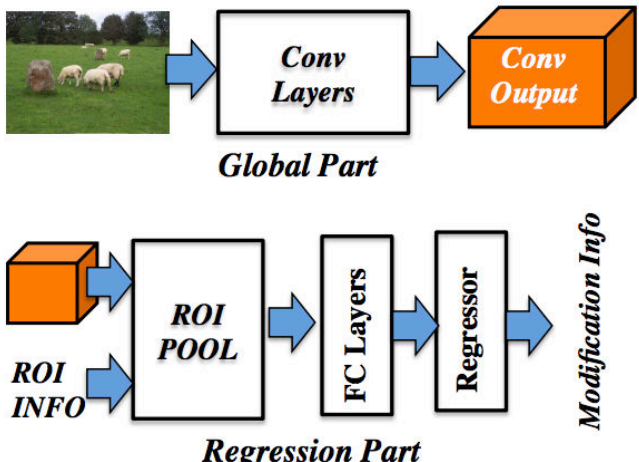
- G-CNN 框的预测目标和损失函数定义

目标：$$\mathcal{A}(\mathbf{B}_i^s) = arg \max_{\mathbf{G} \in \mathcal{G}_i} IoU(\mathbf{B}_i^1, \mathbf{G})$$

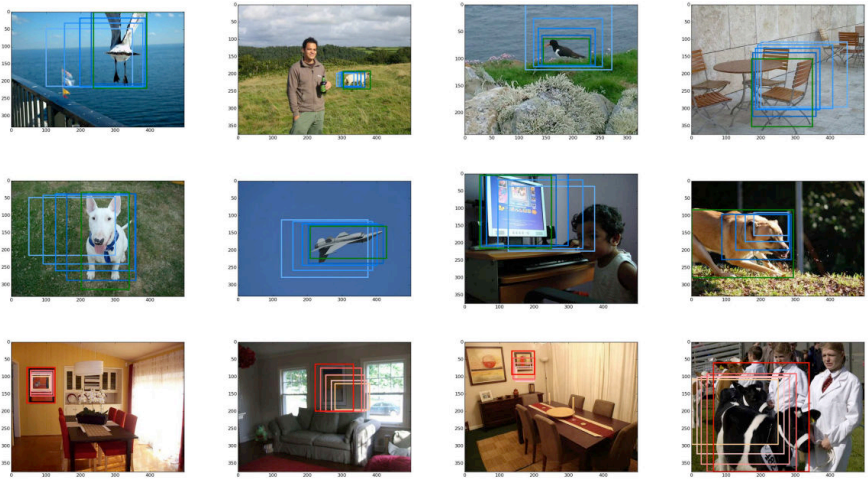
距离定义：$$\Phi(\mathbf{B}_i^s, \mathbf{G}_i^*, s) = \mathbf{B}_i^s + \frac{\mathbf{G}_i^* - \mathbf{B}_i^s}{S_{train} - s + 1}$$

损失函数：$$L(\{\mathbf{B}_i\}_{i=1}^N) = \sum_{s=1}^{S_{train}} \sum_{i=1}^{N} \left[I(\mathbf{B}_i^1 \notin \mathcal{B}_{BG}) \times L_{reg}(\delta_{i,l_i}^s - \Delta(\mathbf{B}_i^s, \Phi(\mathbf{B}_i^s, \mathcal{A}(\mathbf{B}_i^s), s))) \right]$$

- G-CNN 的全局和回归部分
  - 回归部分循环执行



- G-CNN 框查找过程

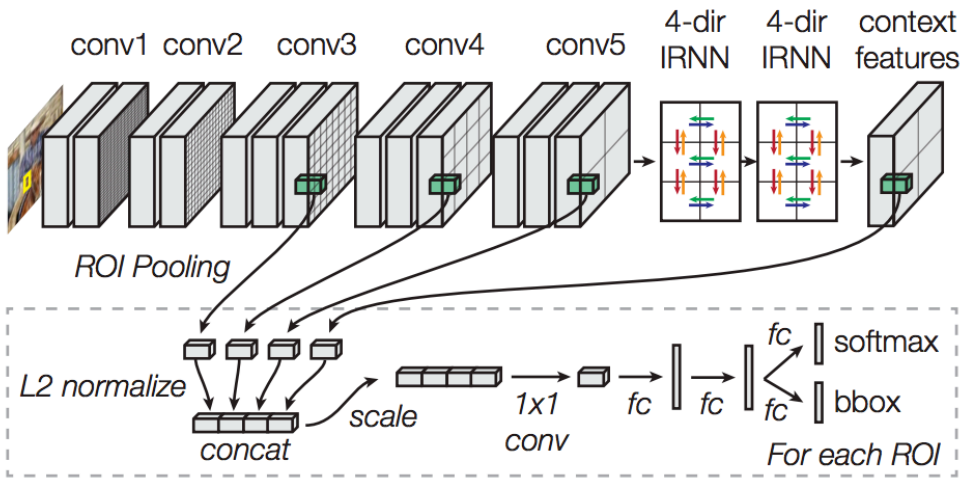


- G-CNN 小结

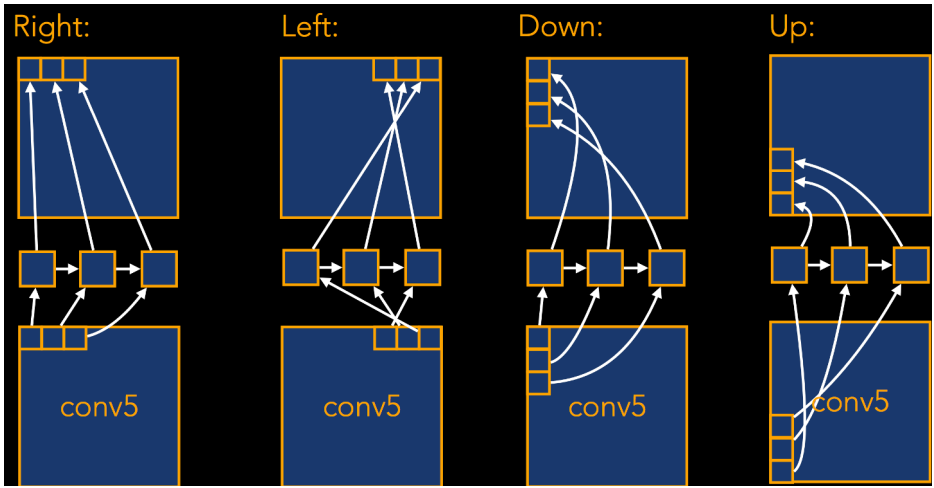
1. 重新定义了框预测的目标：框搜索流程代替框推荐确认流程
2. 比 Fast R-CNN 快
3. 不如 SSD 实时

2.15 ION

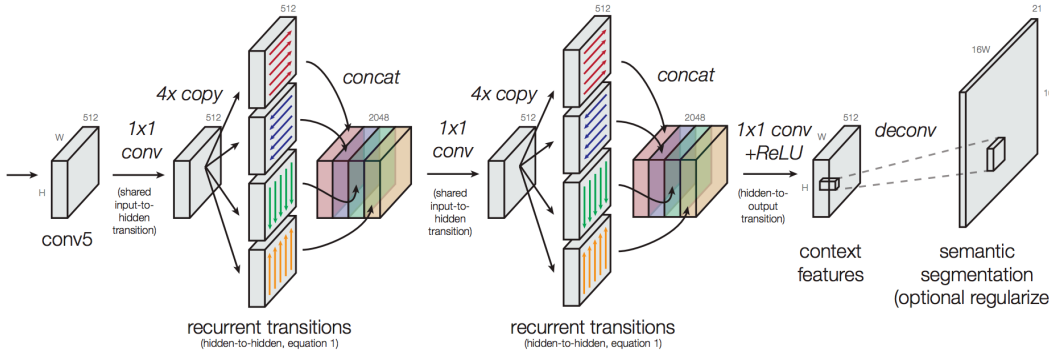
- ION (Inside-Outside Net)



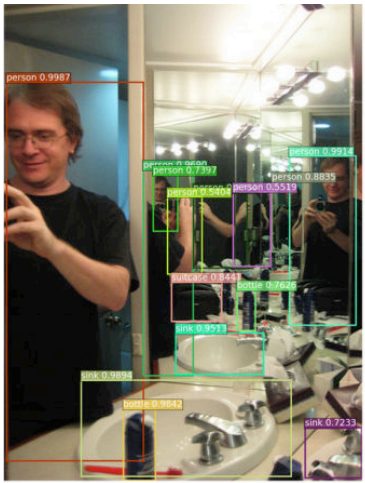
- 四方向 IRNN

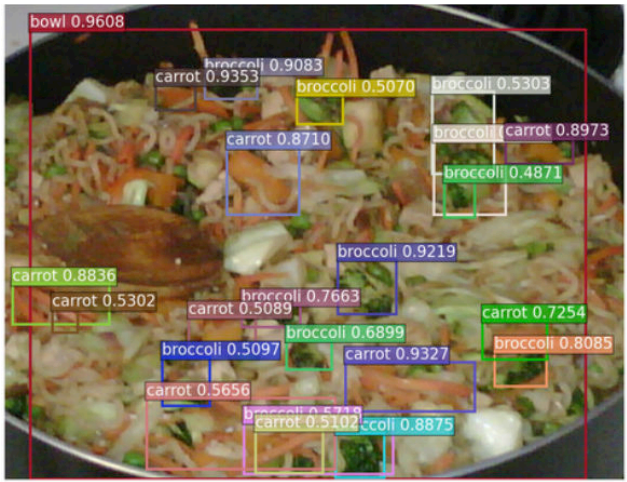


- IRNN 堆栈网络：上下文



- ION 效果：对于小的物体识别优化



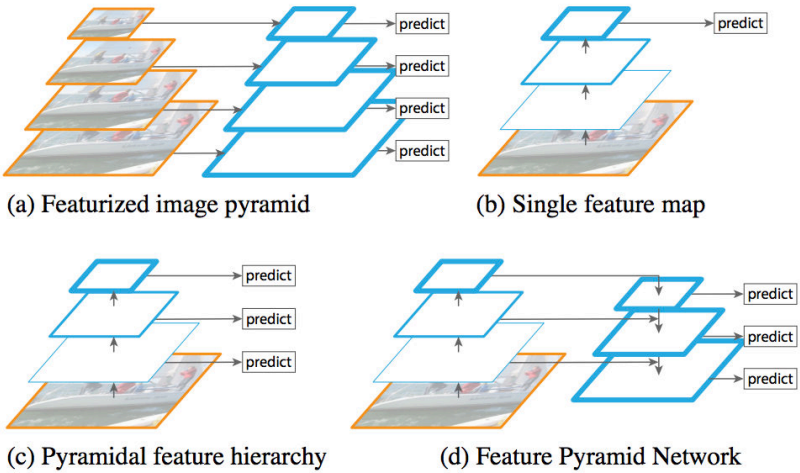


- ION 小结

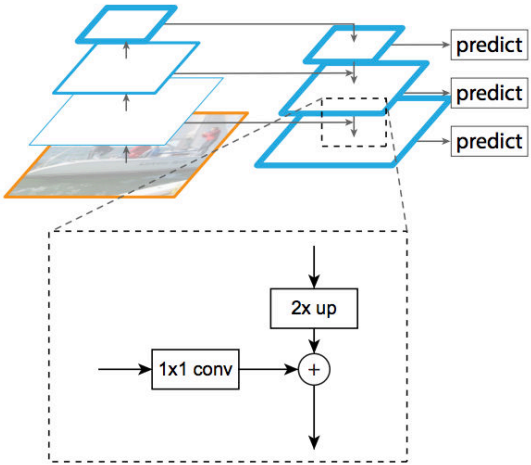
1. 利用 RNN 构建上下文
2. 对小物体识别的优化
3. 比 R-FCN 效果要好

2.16 FPN

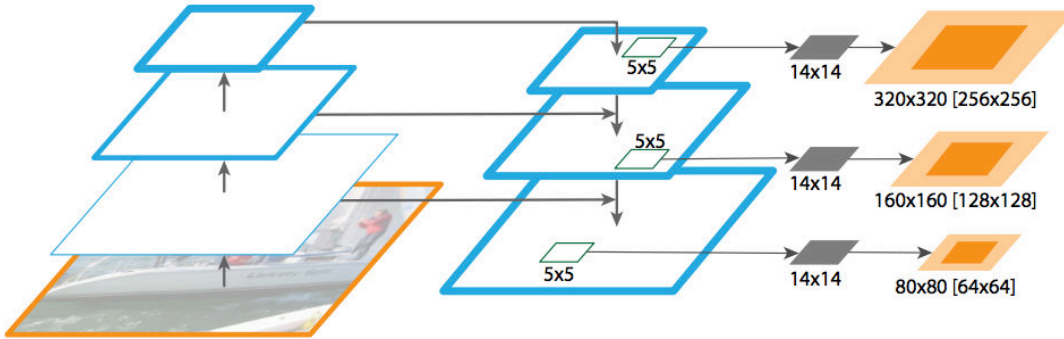
- FPN （Feature Pyramid Networks）



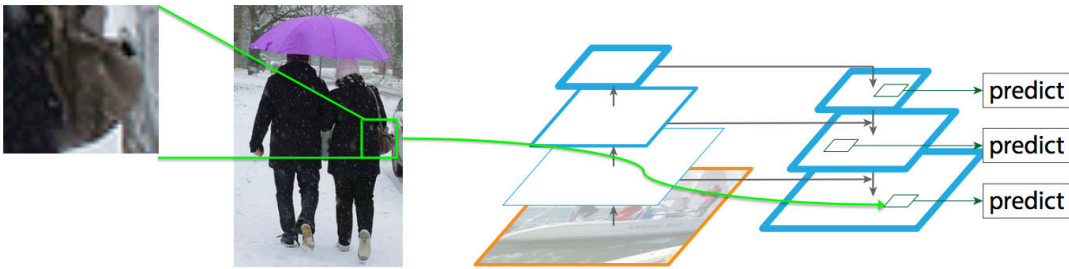
- FPN：自上而下通道，带上下文



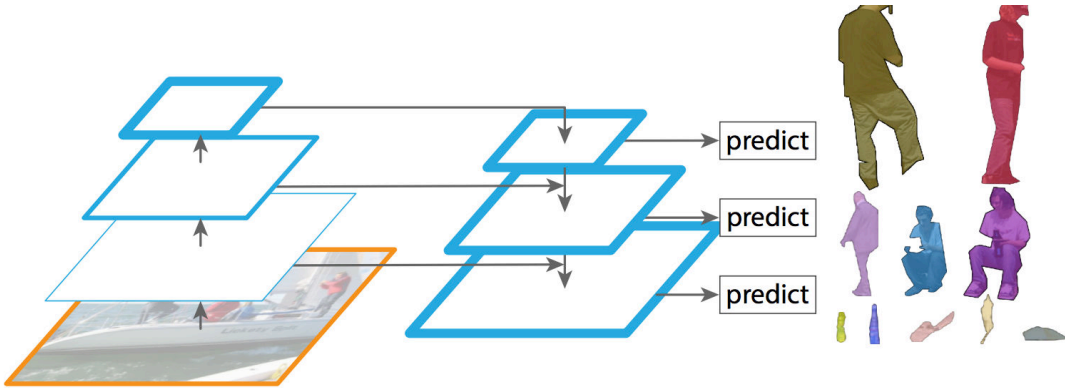
- 分层的分割提议


5x5
5x5
5x5
14x14
14x14
14x14
320x320 [256x256]
160x160 [128x128]
80x80 [64x64]

- 提高小物体特征分辨率



- 多尺度兼容效果



- FPN 效果：速度和准确度的兼容

Strong Features Enable Efficient Learning

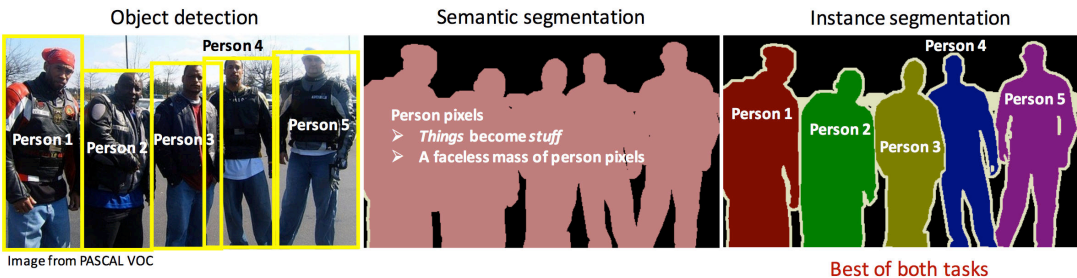
	Fast R-CNN + FPN	Fast R-CNN
Feature dimension	256	1024
Head Classifier	2-mlp	conv5
Training Time	10.6 hr	44.6 hr
Inference Time	0.15 s	0.32 s
Accuracy	33.9 AP	31.9 AP

- FPN 小结

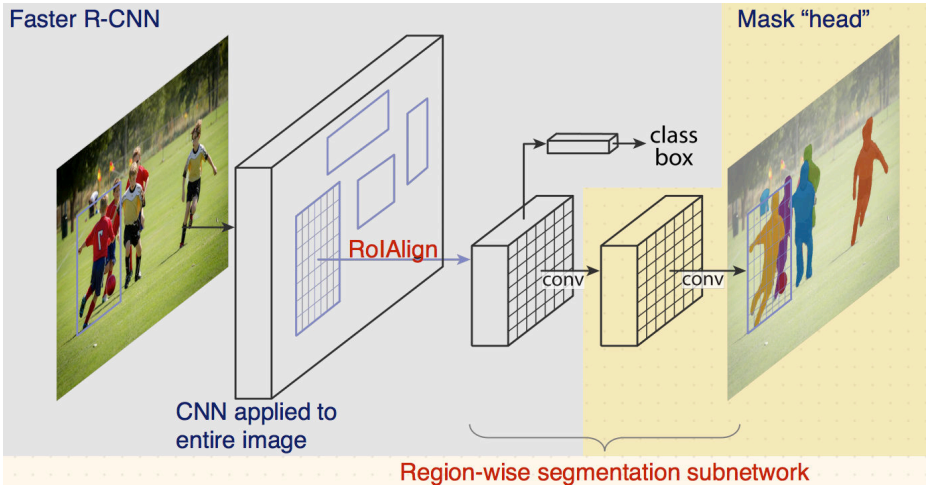
1. 多尺度和小物体的融合考虑
2. 速度和准确率的兼容
3. 可以广泛的结合

2.17 Mask R-CNN

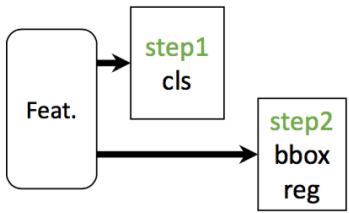
- 实例分割



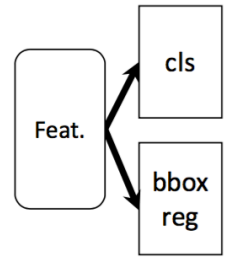
- Mask R-CNN: 对 Faster R-CNN 进一步扩展



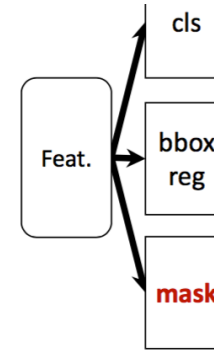
- Mask R-CNN: Faster R-CNN + Mask



(slow) R-CNN

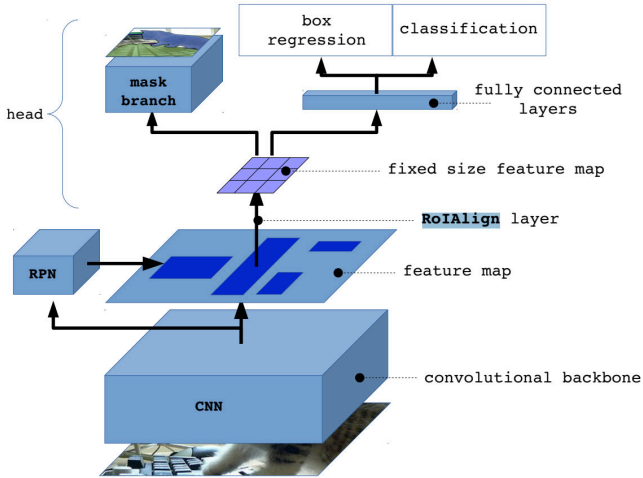


Fast/er R-CNN

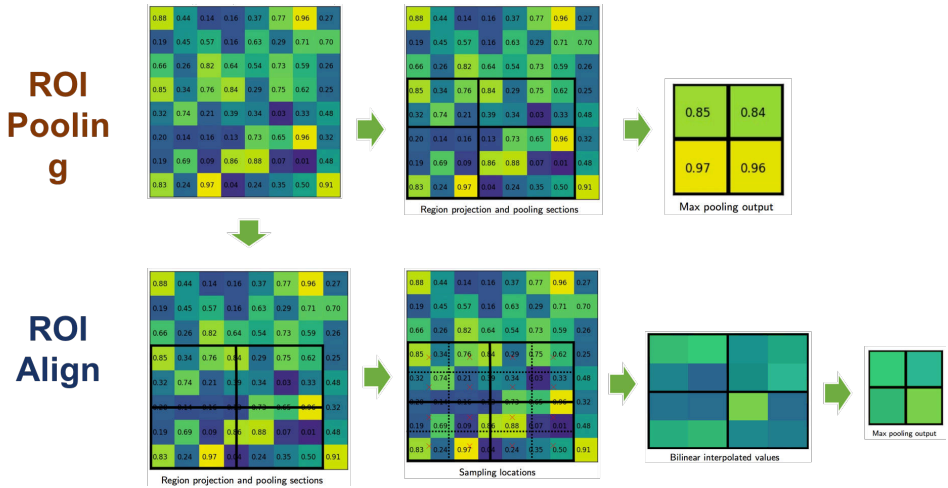


Mask R-CNN

- Mask R-CNN
  - ROIAlign、Mask Branch、



- ROI Align

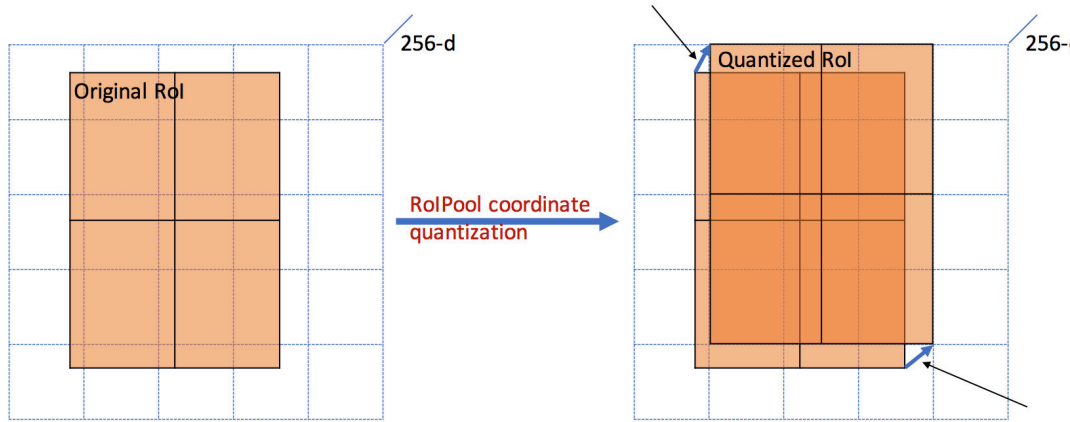

ROI Poolin g
0.85 0.84 0.97 0.96
Max pooling output
Region projection and pooling sections
ROI Align
Region projection and pooling sections
Sampling locations
Bilinear interpolated values
Max pooling output

- ROI Align 效果
  - 严格边界、重新插值调整、最大值池化

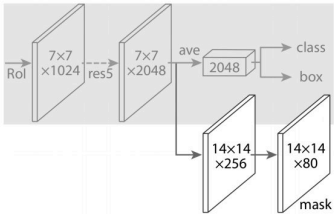
RoIAlign layer:

	stride 16			stride 32*		
	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP
RoIPool	26.9	48.8	26.4	23.6	46.5	21
RoIAlign	**30.2**	**51.0**	**31.8**	**30.9**	**51.8**	**32**

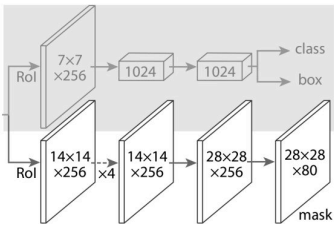
- ROI Pooling 的局限



- Mask R-CNN 两种头部实现

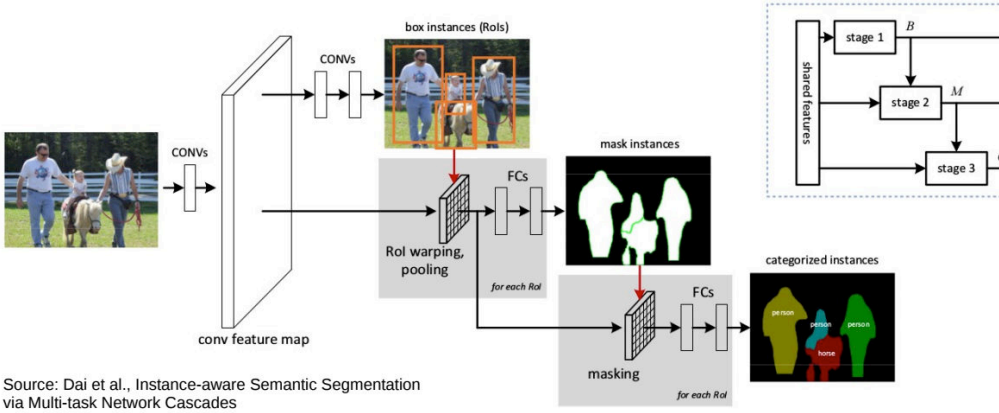


Extended Faster R-CNN head, on ResNet-C4 feature map



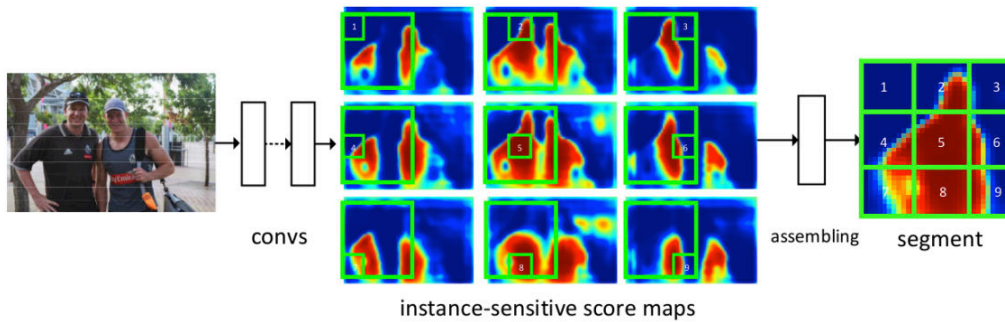
Extended Faster R-CNN head, on FPN feature map

- 来自 MNC(Multi-task Network Cascade) 的启发 (COCO 2015 冠军)
  - Mask 估计、ROI Wrapping 的插值计算

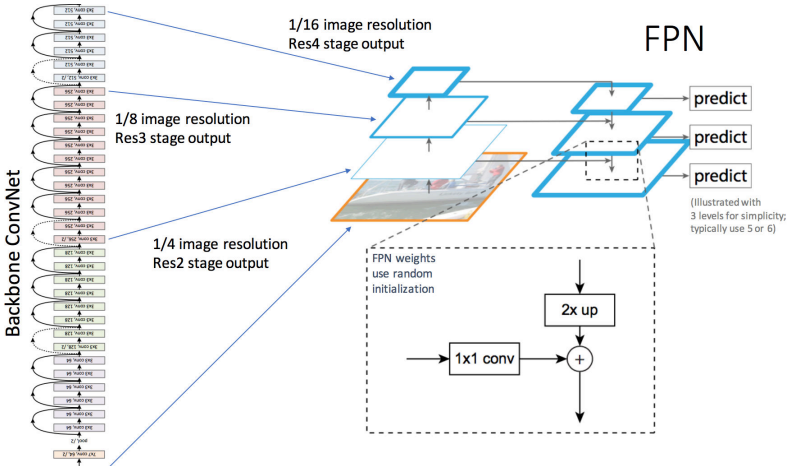


Source: Dai et al., Instance-aware Semantic Segmentation via Multi-task Network Cascades

- 来自 FCIS (Fully Convolutional Instance Segmentation) 的启发 (COCO 2016 冠军)
  - positional aware sliding masks

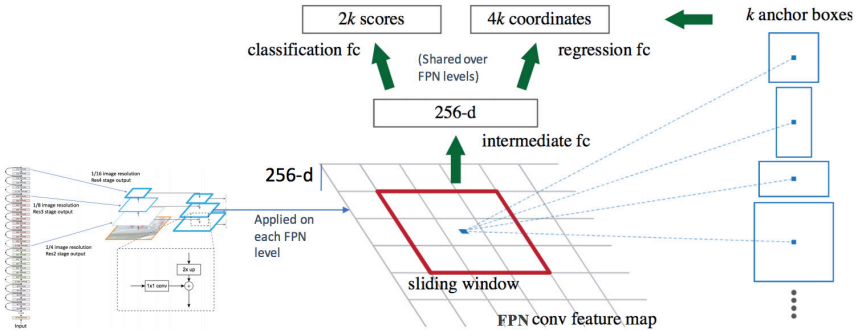


- Mask R-CNN 主干结构：FPN

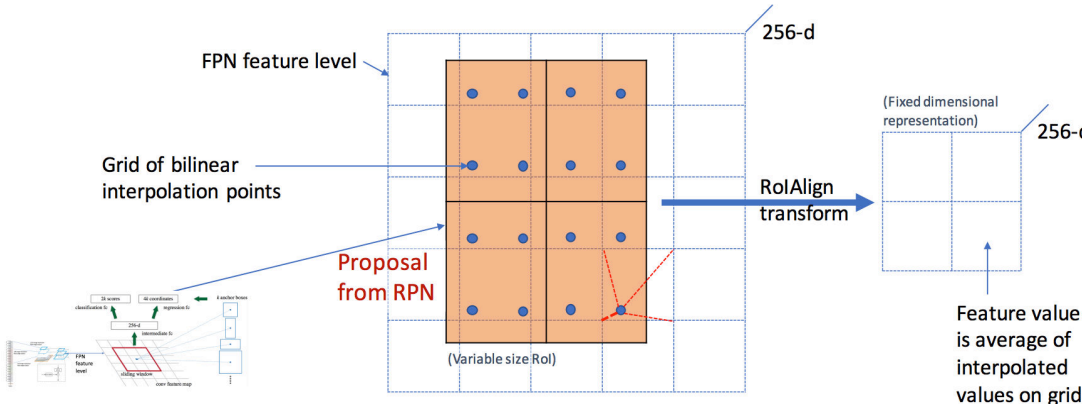


Lin et al. Feature Pyramid Networks for Object Detection. In CVPR 2017. Poster 99 Saturday morning.

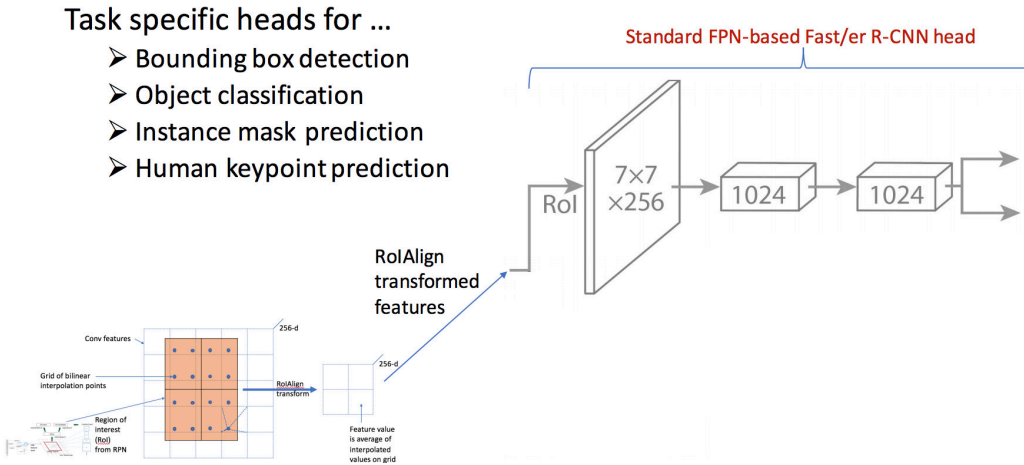
- Mask R-CNN 区域推荐：RPN



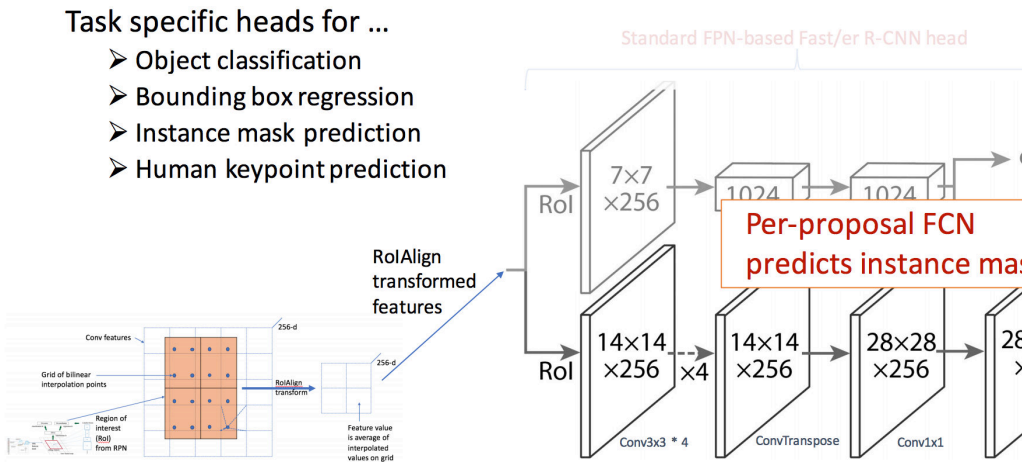
- Mask R-CNN 区域特征：ROIAlign



- Mask R-CNN 头部：具体任务相关
  - 框回归（top 100 NMS ）、物体分类



- Mask R-CNN 实例分割
  - 实例 Mask 预测、人体姿态预测



- Mask 举例

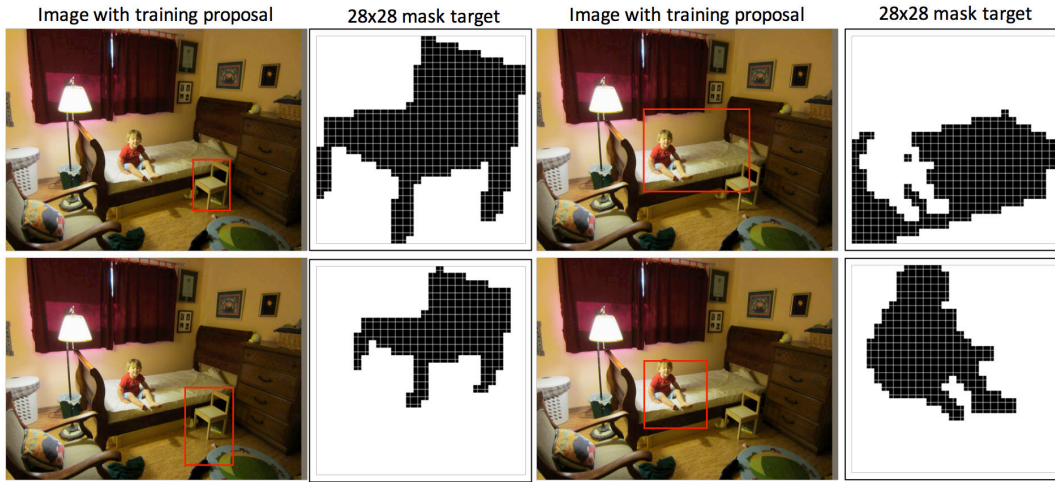
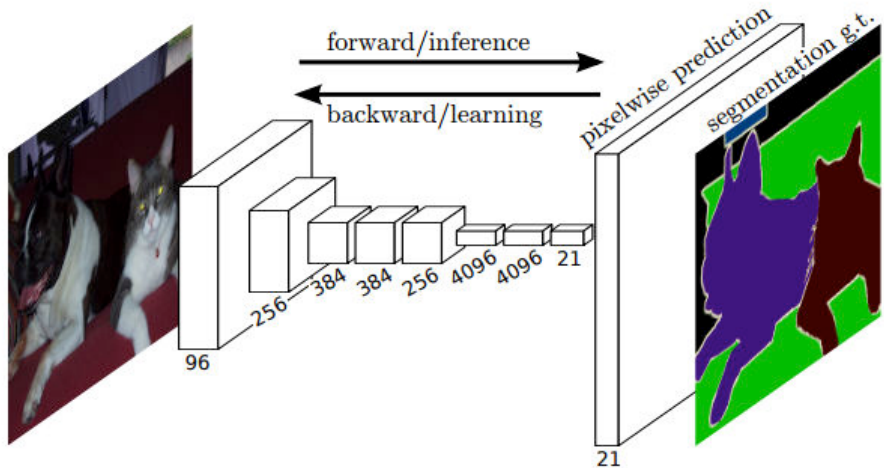
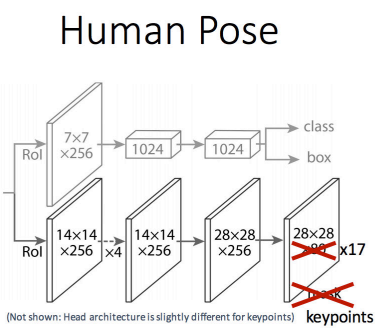


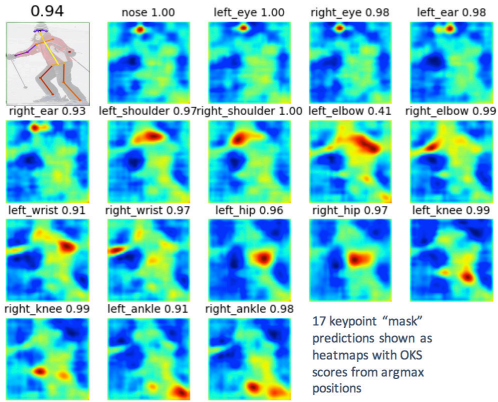
Image with training proposal
28x28 mask target
Image with training proposal
28x28 mask target

- FCN（Fully Convolutional Networks）的 Mask



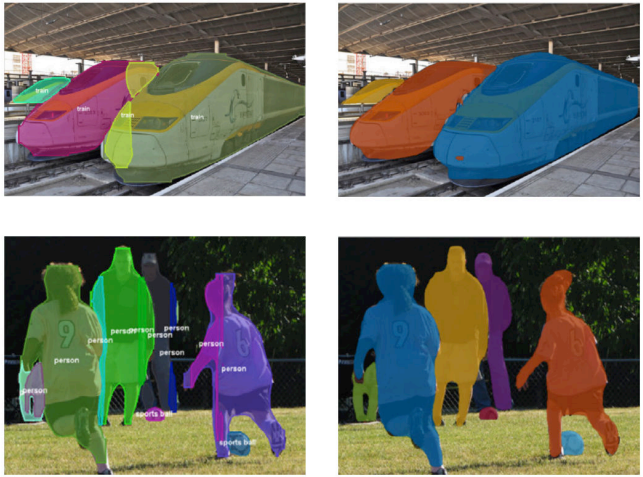
- Mask 人体姿态：17 个要点



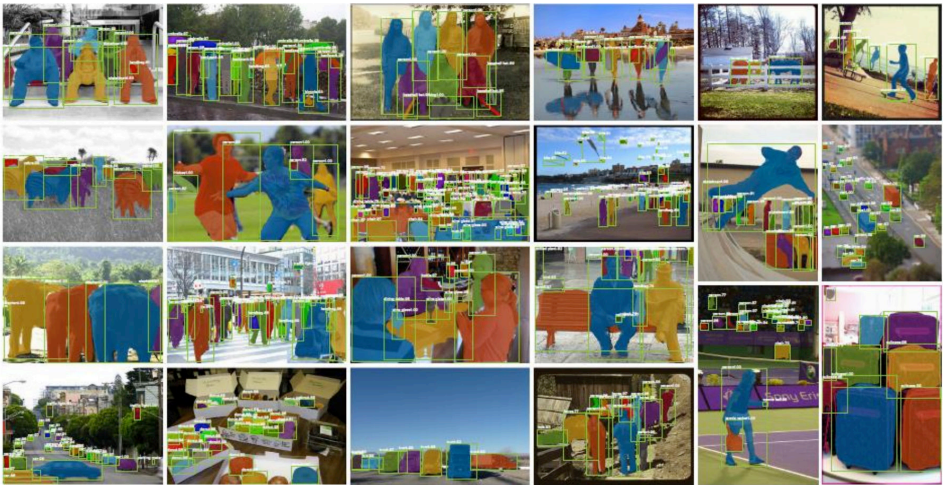


- Add keypoint head (28x28x17)
- Predict one "mask" for each keypoint
- Softmax over spatial locations (encodes one keypoint per mask "prior")

- Mask R-CNN vs FCIS



- Mask R-CNN 结果



- Mask R-CNN：人体姿态标记

- Mask R-CNN 小结

1. ROIPool 到 ROIAlign (借鉴了 ROI Wrapping)
2. Mask 的预测 (借鉴了 MNC 和 FCIS)
3. State-of-Art 的效果
4. 轻微调整可以做人体姿态识别

2.18 图像视频处理小结

- 三大类算法

1. 基于区域推荐
   - R-CNN、Fast R-CNN、Faster R-CNN、Mask R-CNN
2. 基于回归
   - Overfeat、YOLO、YOLO2、SSD
3. 基于搜索
   - AttentionNet、AttratioNet、G-CNN

- 简单选择

1. 速度优先：SSD 算法
2. 速度和效果均衡：R-FCN 算法
3. 效果优先：Faster R-CNN, Mask R-CNN
4. 一网多用：Mask R-CNN

- 经典模块

1. ROI Pool, ROI Wrapping, ROI Align
2. Selective Search, RPN (Region Proposal Networks)
3. HoG Feature Pyramid, FPN (Feature Pyramid Networks)
4. IRNN 上下文
5. NMS (Non-Maxima Suppression)

感谢 Stanford, CMU, MIT 等网上公开课程和资料！

AI2ML