

Lecture 6: 目标检测

目标检测

目标检测与分割分类

概念: 目标检测与分割分类

- 概念: 语义分割
 - 将每一个像素都识别出来是哪一个物体的一部分;

语义分割
(Semantic Segmentation)



GRASS, CAT,
TREE, SKY

- 概念: 目标定位与识别
 - 在图片当中只有一个待识别的物体，需要给出在图片当中、物体的位置，并给出对应的区域;

目标定位与识别
(Classification and Localization)



CAT

- 概念: 目标检测
 - 在图片中有多个物体，将其分出来并且分类;

目标检测 (Object Detection)



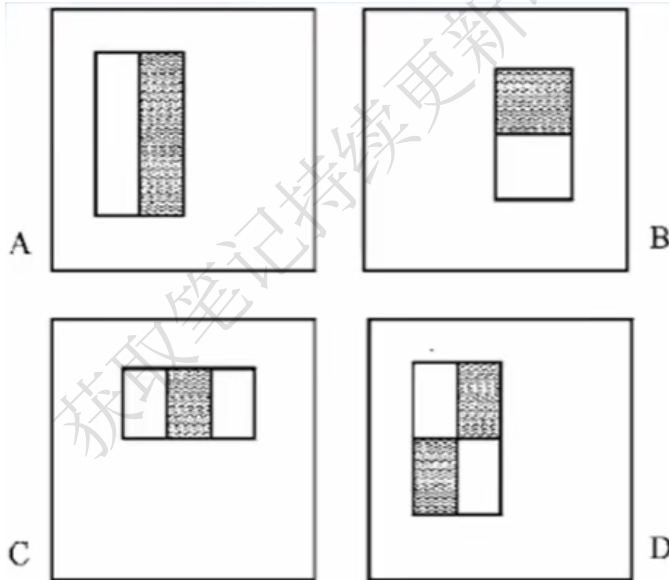
DOG, DOG, CAT

- 技术：目标分割
 - 无论是哪种，都需要使用目标分割，将需要的数据和不需要的数据分开；
- 技术：方框扫描
 - 从小方框到大方框，逐个进行遍历扫描；
 - 效率低；

AdaBoost 与目标检测

概念：AdaBoost 与目标检测

- 概念：



Haar-like Features

(1) 白色区域的像素值，减去黑色区域的像素值。

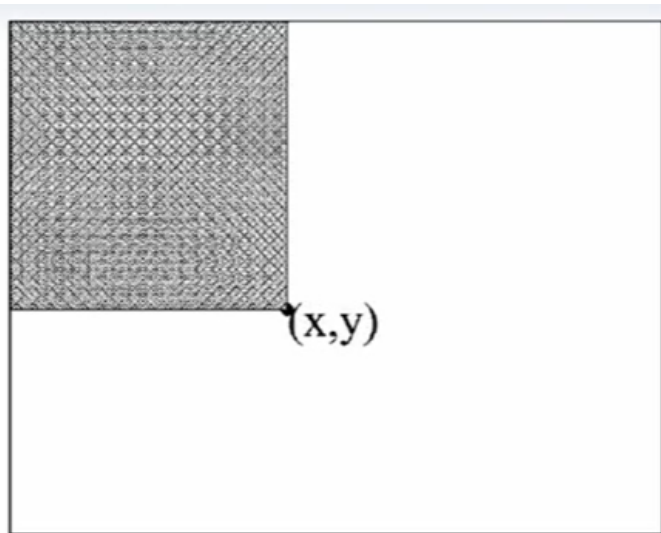
(2) 每一个FEATURE所有区域长度和宽度一致。

(3) FEATURE可以在整幅图上平移，只需要满足(1)和(2)即可。

(4) 可以取左图这四种形式。

对于一个 24×24 的图像，所有的Haar-like feature 个数为20万左右。

- 概念：积分图
 - 在 (x, y) 点的图象值是二重积分的结果，即求和：



$$\text{Integral}(x,y) = \sum_{u \leq x} \sum_{v \leq y} \text{Image}(u,v)$$

- 所以这个平面是左上角所有像素的求和；
- 这个积分图得到后，就可以将特征图的提取用加减法描述；

• 概念：分类器构造

- 分类器构造：取一些人脸 (6000 张左右) 和一些非人脸 (7 万张) 作为训练样本；

$$h(x, f, p, \theta) = \begin{cases} 1 & \text{if } p f(x) < \theta \\ 0 & \text{otherwise} \end{cases}$$

- x: 图像；
- f: 某一个 Haar-like feature；
- θ : 阈值；
- p: POLARITY, 只能取+1 或-1；

- 对某个特定的 f, 求最佳的 θ, p 取值, 使 h 在训练样本上识别率最高；

• 总结：

- (1) 首先在数据集 D 中选取正确率最高的特征用 F1 表示；
- (2) 将数据集 D 分为两类, {F1 分对的数据} 和 {F1 分错的数据}；
- (3) 以较大概率取 F1 分错的数据, 以较小概率去 F1 分对的数据, 形成新的集合 D2；
- (4) 在 D2 中选取正确率最高的特征, 用 F2 表示；
- (5) 将 D 分为: {F1、F2 都分对的数据}, {F1 分对而 F2 分错的数据, 以及 F1 分错而 F2 分对的数据}, {F1、F2 都分错的数据}；
- (6) 以最大概率取 {F1、F2 都分错的数据}, 以次大概率取 {F1 分对而 F2 分错的数据, 以及 F1 分错而 F2 分对的数据}, 以最小概率取 {F1、F2 都分对的数据}, 得到数据集 D3；
- (7) 在 D3 中选取正确率最高的特征, 用 F3 表示；
- (8) 循环, 以此类推；
- (9) 用各个特征的线性组合构建分类器；

• 图示：

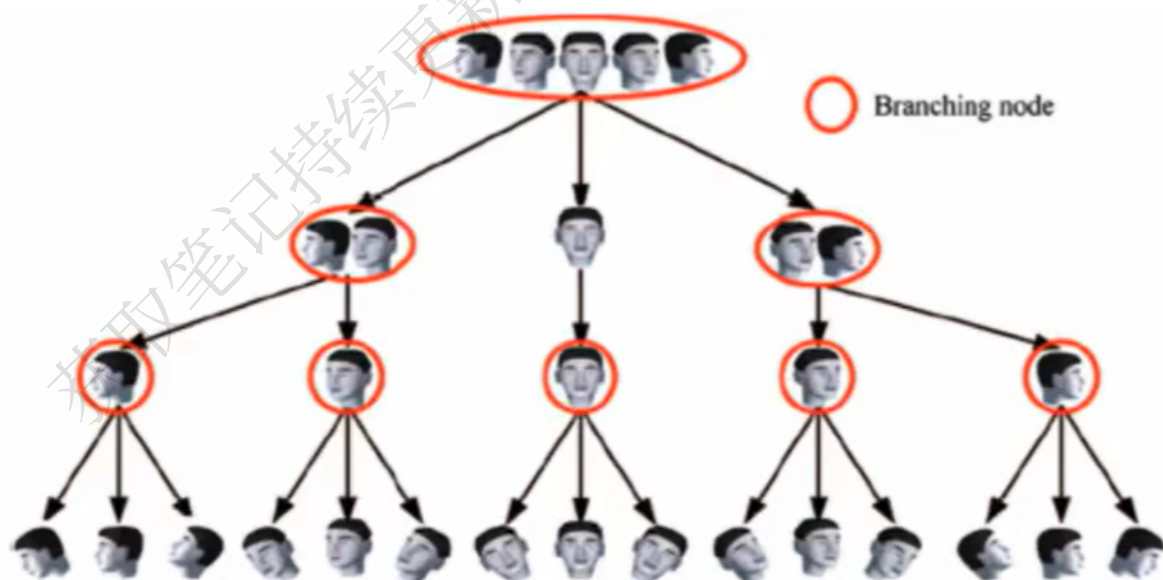
- 最能表征人脸的 Haar 特征;



AdaBoost 人脸检测流程:

1. 在图像中，对每一个 24×24 的格子遍历使用分类器，如果是人脸，则输出。
2. 将图像缩小，长宽同时除以1.2，在用分类器遍历每一个 24×24 的格子。如果是人脸，将该处位置坐标乘以1.2，等比例放大到原图。
3. 重复2，直到图像长或宽小于24个像素为止。

- 补充:



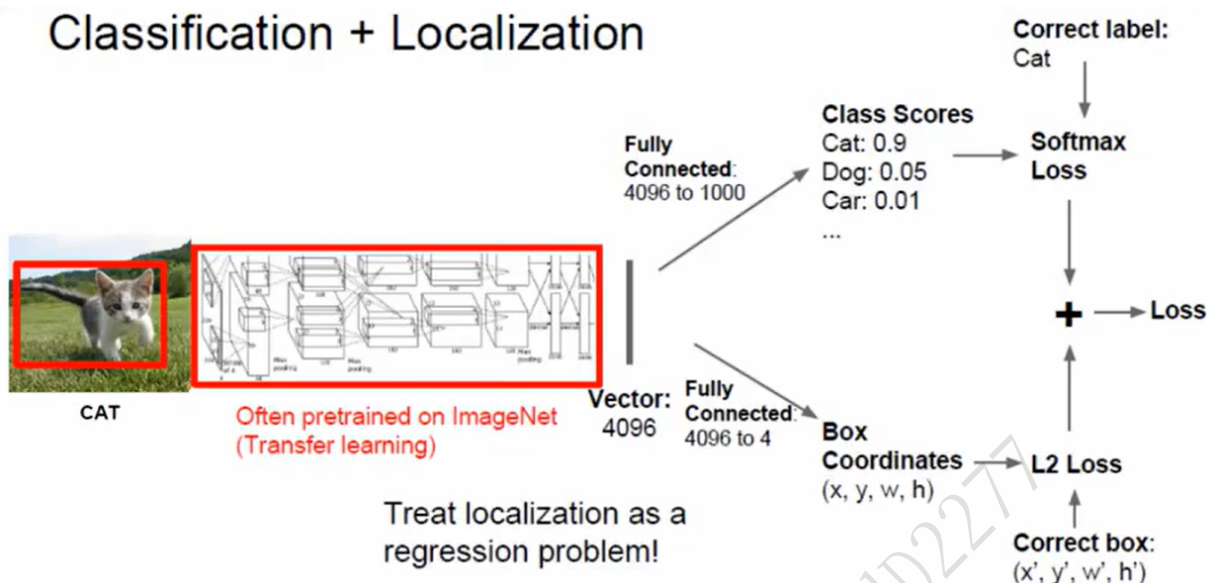
多个姿态人脸都可以构建AdaBoost分类器，树状级联后可以获得多姿态人脸检测器

基于卷积神经网络进行目标检测

分类与定位问题

概念：基于卷积神经网络进行分类与定位问题

- 图示：



- 分析：

- 在一个图形中有且仅有一个对象，这是一个多分类的问题，需要得到四个坐标，将对象框住；

分类检测问题

概念：基于卷积神经网络进行分类检测

- 问题：对于一个图像中，有很多个对象需要检测，因此需要不同的设计；

目标检测

基础概念

概念：边缘框

- 概念：
 - 一个边缘框可以通过 4 个数字定义：
 - 方式一：左上 x 左上 y, 右下 x, 右下 y;
 - 方式二：左上 x, 左上 y, 宽, 高;
- 概念：目标检测的数据集
 - 每行表示一个物体
 - 图片文件名, 物体类别, 边缘框
 - COCO cocodataset.org
 - 80 物体, 330 K 图片, 1.5 M 物体;

代码：目标检测和边缘框

```
import torch
from d2l import torch as d2l
```

```

d2l.set_figsize()
img = d2l.plt.imread('../img/catdog.jpg')
d2l.plt.imshow(img)

# 边界框
def box_corner_to_center(bboxes):
    x1,y1,x2,y2 = bboxes[:,0],bboxes[:,1],bboxes[:,2],bboxes[:,3]
    cx = (x1 + x2)/2
    cy = (y1 + y2)/2
    w = x2-x1
    h = y2-y1
    bboxes = torch.stack((cx,cy,w,h),axis=-1)
    return bboxes

# 画出边界框
def bbox_to_rect(bbox,color):
    return d2l.plt.Rectangle(xy=(bbox[0],bbox[1]),width=bbox[2]-bbox[0],height=bbox[3]-bbox[1],fill=False,edgecolor=color,linewidth=2)

```

锚框

概念：锚框

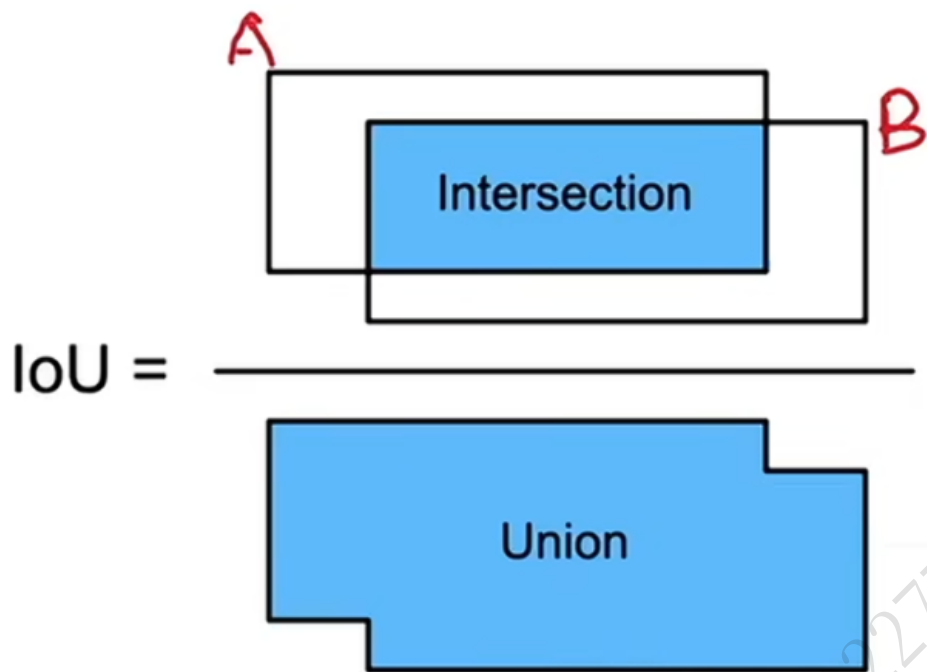
- 概念：
 - 算法会提出多个被称为锚框的区域（边缘框），预测每个锚框里是否含有关注的物体；
 - 如果是，预测从这个锚框到真实边缘框的偏移，将其慢慢调整到目标的位置；

• 概念：IoU - 交并比

- IoU 用来计算两个框之间的相似度
 - 0 表示无重叠，1 表示重合
- 假设给定两个集合 A 和 B：

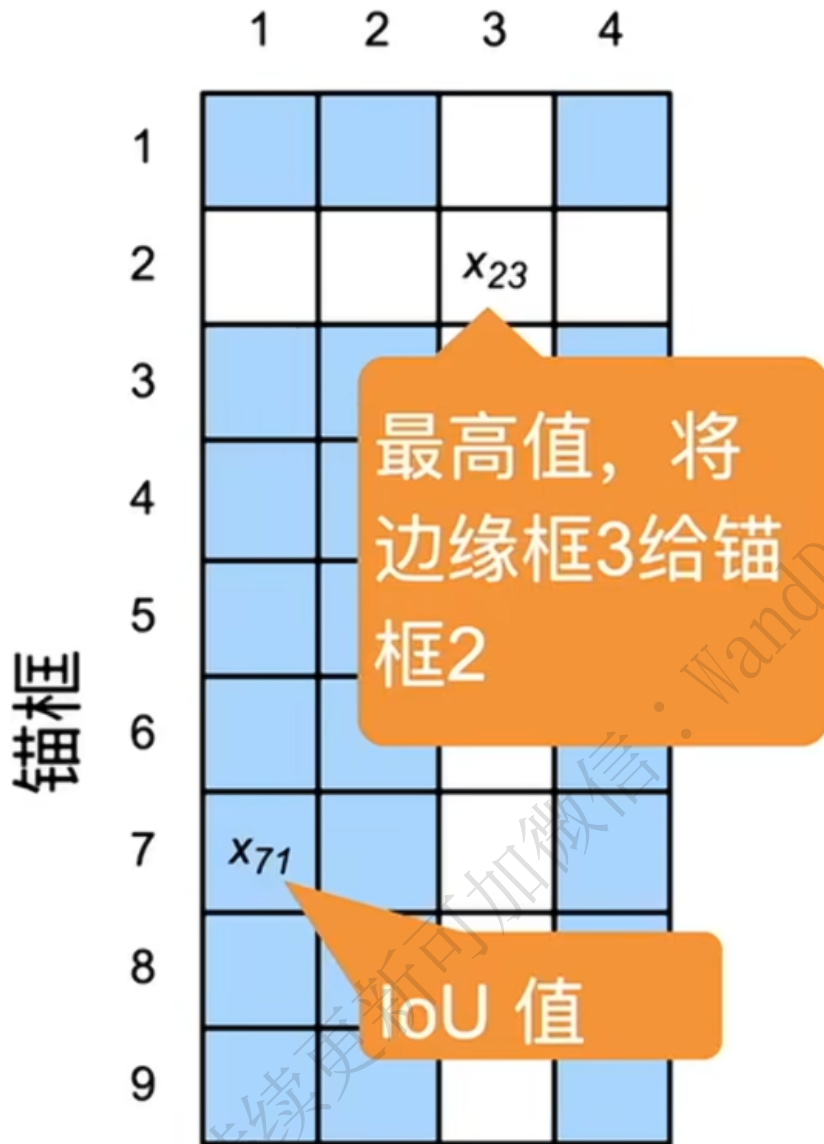
$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

- 图示：



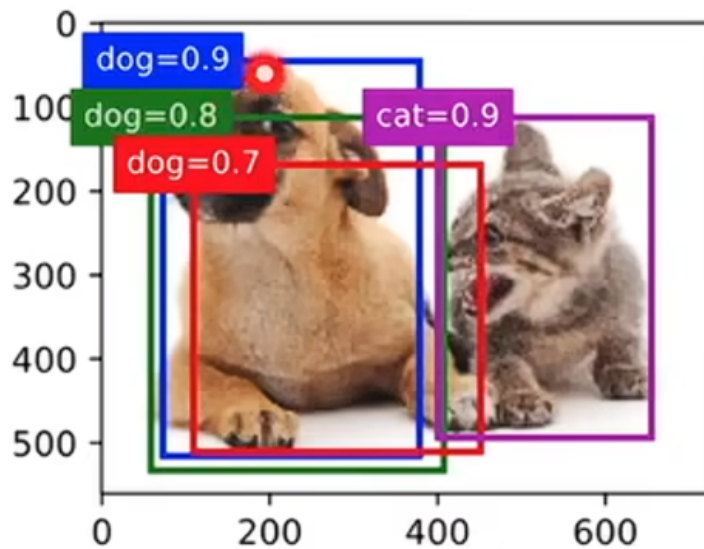
- 概念：给锚框标号
 - 概念：
 - 每个锚框是一个训练样本将每个锚框，其要么标注成背景，要么关联上一个真实边缘框；
 - 因此：一个算法可能可能会生成大量的锚框，绝大部分的锚框都是背景，因此导致大量的负类样本，所以目标检测算法需要处理这些负类样本；
 - 图示：

边缘框



概念：使用非极大值抑制（NMS）输出

- 概念：
 - 每个锚框预测一个边缘框，因此可能一张图片会有很多个锚框；
 - NMS 可以合并图中相似的预测；
- 实现：
 - (1) 选中所有框当中、对应的类别预测值中、非背景类的最大预测值；



- (2) 去掉所有其它和它 IoU 值大于 θ 的预测;
- (3) 重复上述过程直到所有预测要么被选中, 要么被去掉

RCNN

概念: RCNN

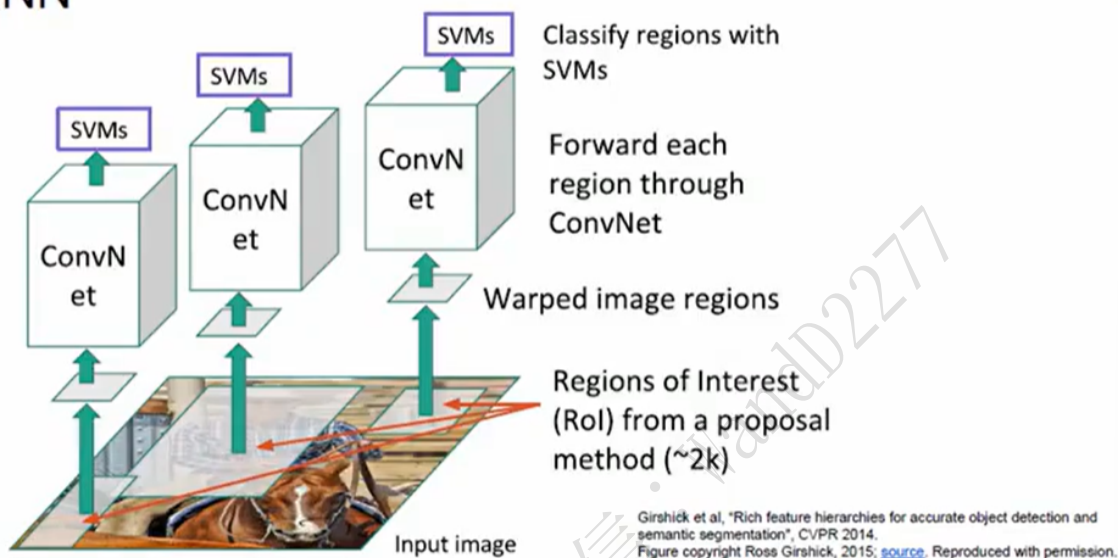
- 问题:
 - 如何将卷积神经网络 (CNN) 用在目标检测上?
- 概念:
 - 单目标检测有一个确定的结果, 而多目标检测不知道目标在哪里;
 - 因此使用 RCNN、可以选出一些目标的候选区域, 即先用一些传统的机器学习方法进行区域分割;
 - 分割完成后、对于每一个候选项, 送到 CNN (卷积神经网络) 中进行识别, 再进行位置的提取;
- 实现:
 - (1) 使用启发式搜索算法来选择锚框;
 - (2) 使用预训练的模型 (比如给 VGG) 来对每个锚框提取特征;
 - (3) 训练一个 SVM 来对类别进行分类;
 - (4) 训练一个线性回归模型来预测边缘框的偏移;
- 过程:
 - 1. 使用 selective search 取生成分割图形;
 - 给定一张图片, 首先使用 Efficient Graph-Based Imagesegmentation 算法, 将图片进行过分割 (Over-Segmentation);



- 如图所示，过分割后的每个 **region** 非常小，以此为基础，对相邻的 **region** 进行相似度判断并融合，形成不同尺度下的 **region**。每个 **region** 对应一个 **bounding**；
- 2. 对于分割的结果使用 **CNN** 进行检测；
- 3. 再使用 **SVM** 做出最终的决定；

• 图示：

R-CNN



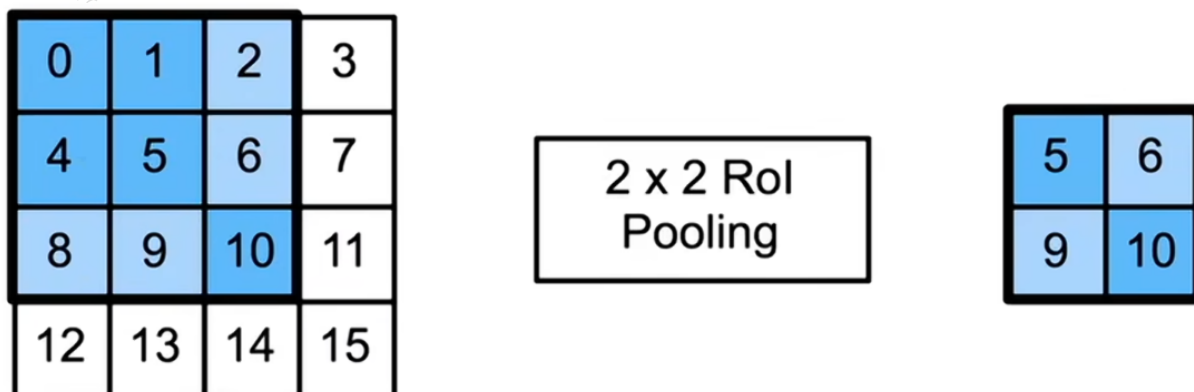
Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." CVPR. 2014.

- 问题：
 - (1) 生成出很多个区域都要送进卷积中，因此效率十分慢，处理一张图可能需要一分钟；
 - (2) 分割的区域过小，而且很多卷积的操作重复；

概念： RoI 池化层

- 概念：
 - 给定一个锚框，均匀分割成 $n \times m$ 块，输出每块里的最大值；
 - 不管锚框多大，总是输出 nm 个值；

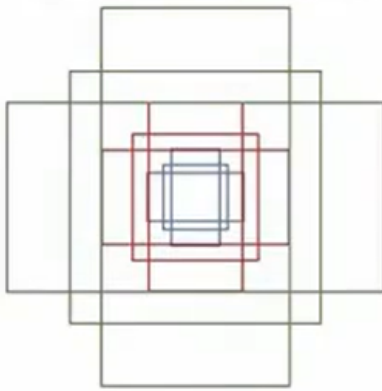
• 图示：



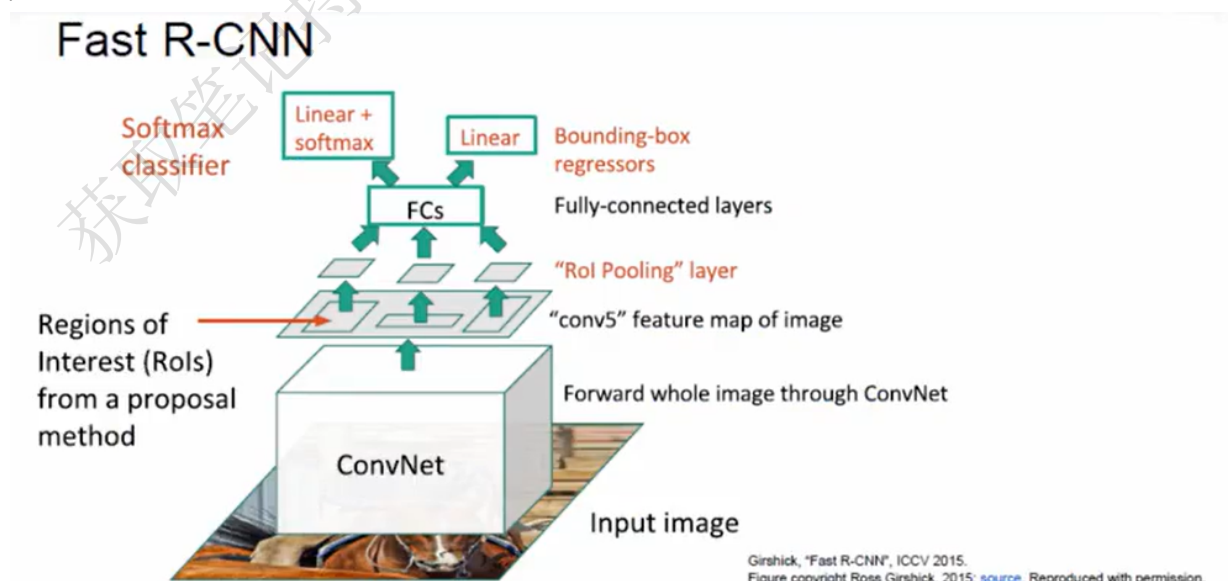
Fast RCNN

概念: Fast RCNN

- 作用:
 - 解决了 RCNN 的大量重复问题;
- 实现:
 - 1. 先将整张图像输入到一个只有几层的 CNN 中, 把中间的某一部的特征图提取出来;
 - 即: 对整个图像输入到卷积层;
 - 2. 进行一次 Pooling, 得到各个区域的感受野;
 - 即特征图上的 Pooling 得到的区域可以对应为图片上的区域;
 - 即在中间的某一层预测;
- 概念:
 - 对于特征图某个固定点, ANCHOR 生成 9 个矩形, 共有 3 种形状长宽比为大约为:
width:height=[1:1, 1:2, 2:1] 三种, 实际上通过 anchors 就引入了检测中常用到的多尺度方法;



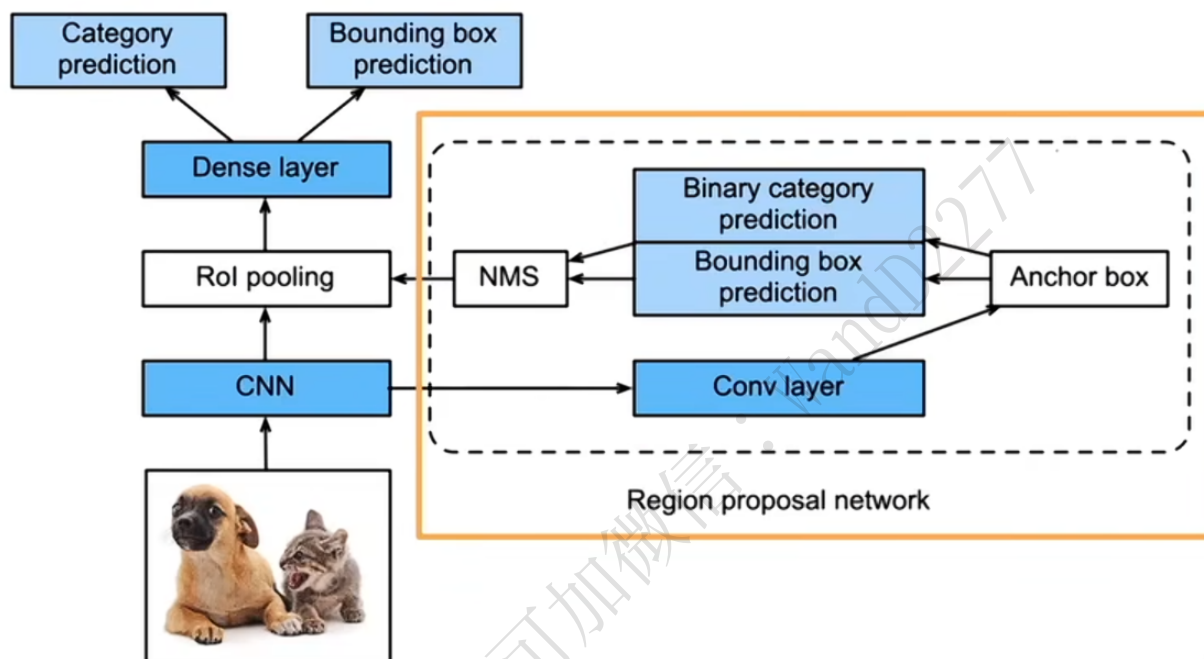
- 把任意大小的输入图像 reshape 成 800×600 (即图 2 中的 $M=800$, $N=600$);
- 再回头来看 anchors 的大小, anchors 中 长宽 1:2 中最大为 352×704 , 长宽 2:1 中最大 736×384 , 基本是 cover 了 800×600 的各个尺度和形状;
- 图示:



Faster R-CNN

概念: Faster R-CNN

- 概念：
 - 使用一个区域提议网络来替代启发式搜索，来获得更好的锚框；
- 概念：Region proposal network
 - 在对 CNN 得到的卷积结构后，将现在得到的比较粗糙的锚框、再对其进行一个二分类；
 - 通过这个神经网络，为后生成更好的锚框；
 - 好的锚框进入 RoI 的池化层；
 - 可以理解为做了两次预测；
- 架构：



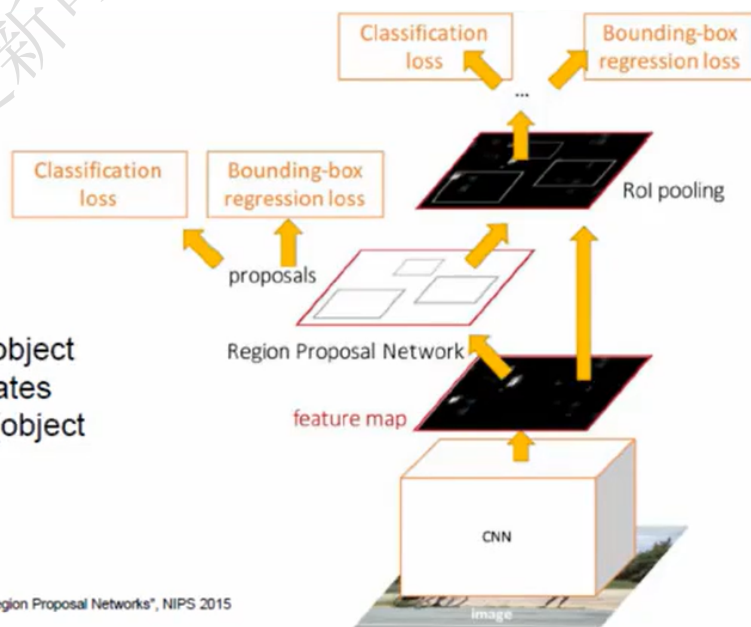
Faster R-CNN:

Make CNN do proposals!

Insert **Region Proposal Network (RPN)** to predict proposals from features

Jointly train with 4 losses:

1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates



Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

总结

- R-CNN 是最早、也是最有名的一类基于锚框和 CNN 的目标检测算法；
- Fast/Faster R-CNN 持续提升性能；
- Faster R-CNN 和 Mask R-CNN 是在最求高精度场景下的常用算法；

YOLO

概念: YOLO

- 概念:
 - SSD 中锚框大量重叠, 浪费了很多的计算;
 - 相邻像素之间生成的锚框十分的重叠;
 - 因此: YOLO 将图片均匀分成 $S \times S$ 个锚框;
 - YOLO 中生成的锚框全部不相交, 也不会浪费计算量;
 - 每个锚框预测 B 个边缘框;
 - 因为有可能单个边缘框中、有好几个物体;
 - 因此其代价就是 $S * S * B$
-

获取笔记持续更新可加微信: WandD2277