深度学习与迁移学习

寒小阳
2017-07-30

主要内容

- 图像识别与定位
  - 思路1：视作回归
  - 思路2：借助图像窗口
- 物体识别
  - 边缘策略/选择性搜索 => R-CNN
  - R-CNN => Fast R-CNN
  - Fast R-CNN => Faster R-CNN
  - R-FCN简介
- 有监督到有监督的迁移学习
  - fine-tune 再优化
  - Multitask learning 多任务学习
- 有监督到无监督的迁移学习
  - 域对抗学习

卷积神经网络

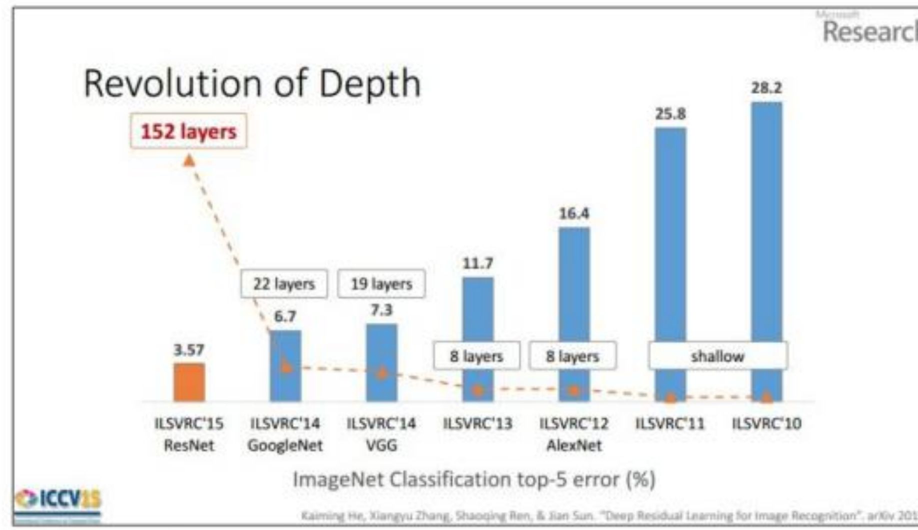
- 层次变深，效果变好

D	E
16 weight layers	19 weight layers
conv3-64 conv3-64	conv3-64 conv3-64
conv3-128 conv3-128	conv3-128 conv3-128
conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool	
FC-4096	
FC-4096	
FC-1000	
soft-max	

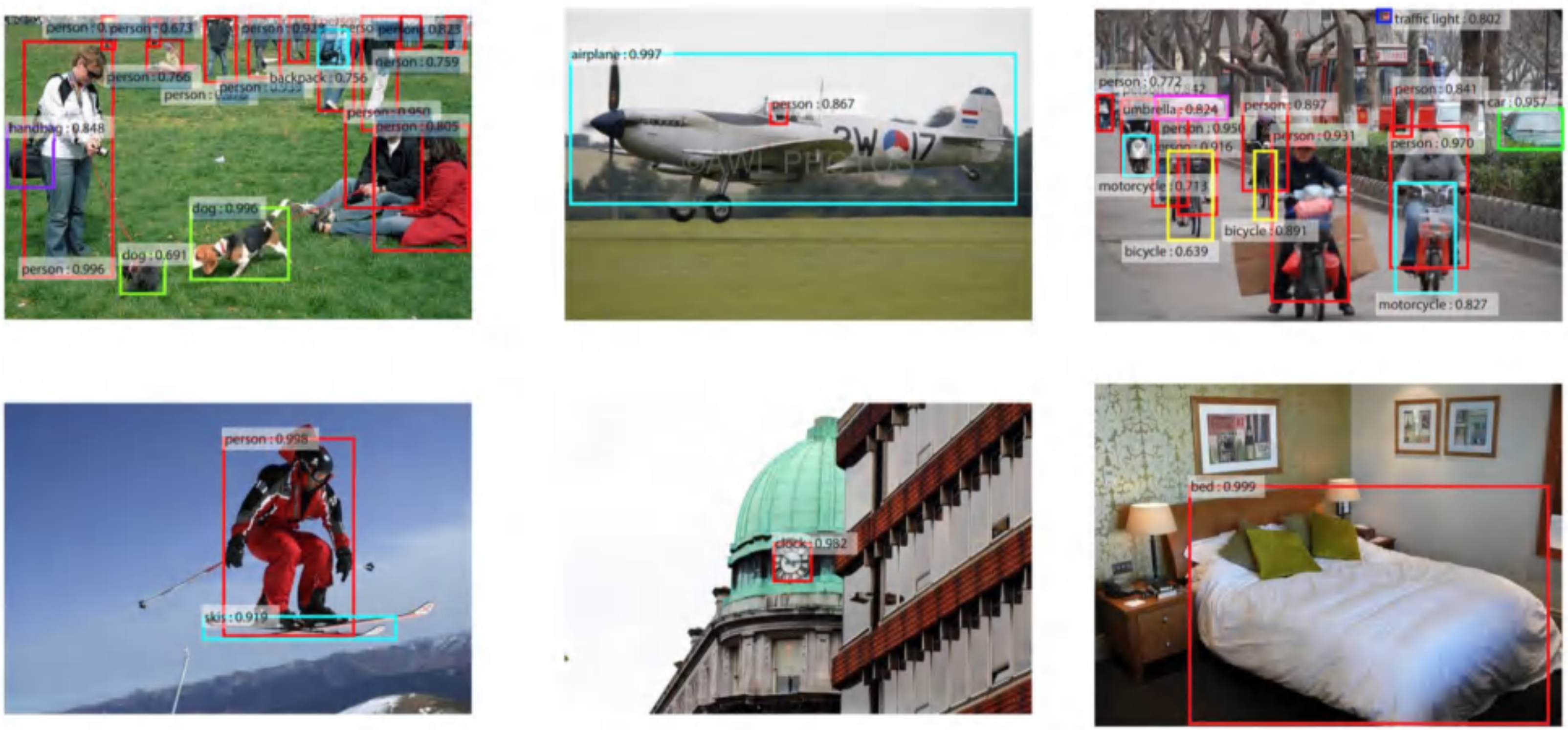
VGG (2014)

GoogLeNet (2014)

ResNet (2015)

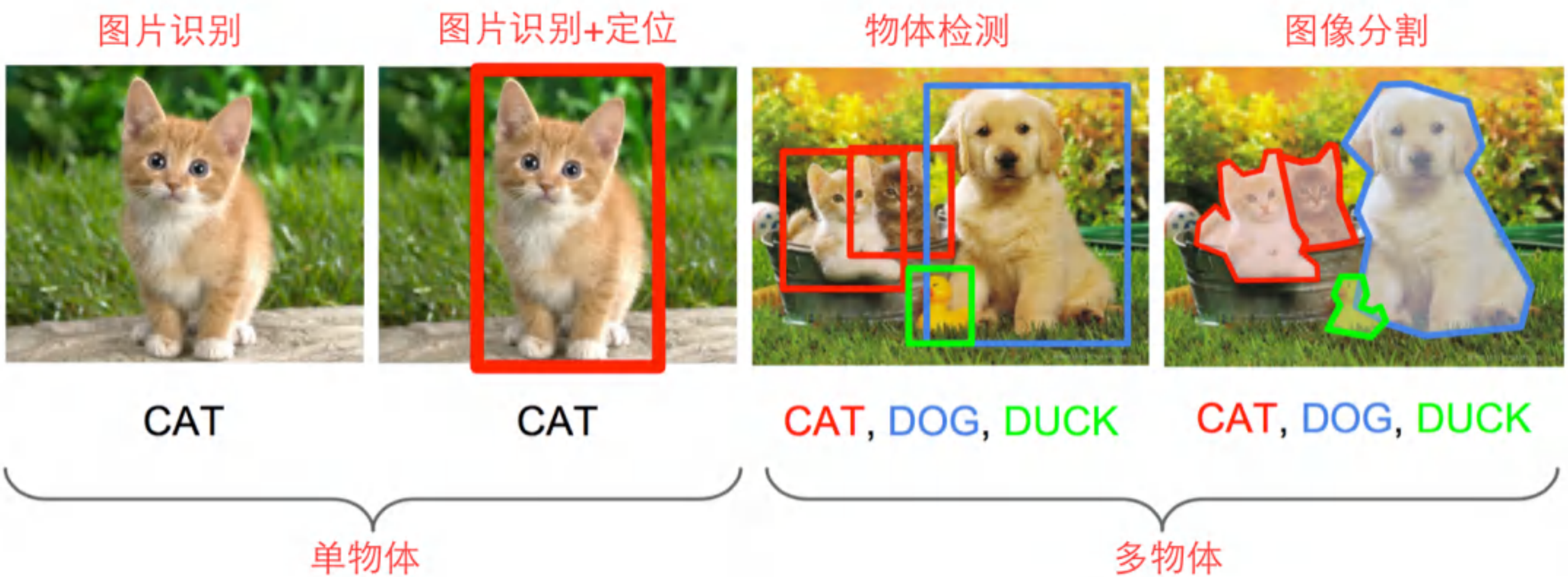


物体识别



Results from Faster R-CNN, Ren et al 2015

图像相关任务



图片识别
图片识别+定位
物体检测
图像分割
CAT
CAT
CAT, DOG, DUCK
CAT, DOG, DUCK
单物体
多物体

图像识别+定位

图像识别与定位

Classification: C个类别

- **Input:** Image
- **Output:** 类别标签
- **Evaluation metric:** 准确率

→ CAT

Localization:

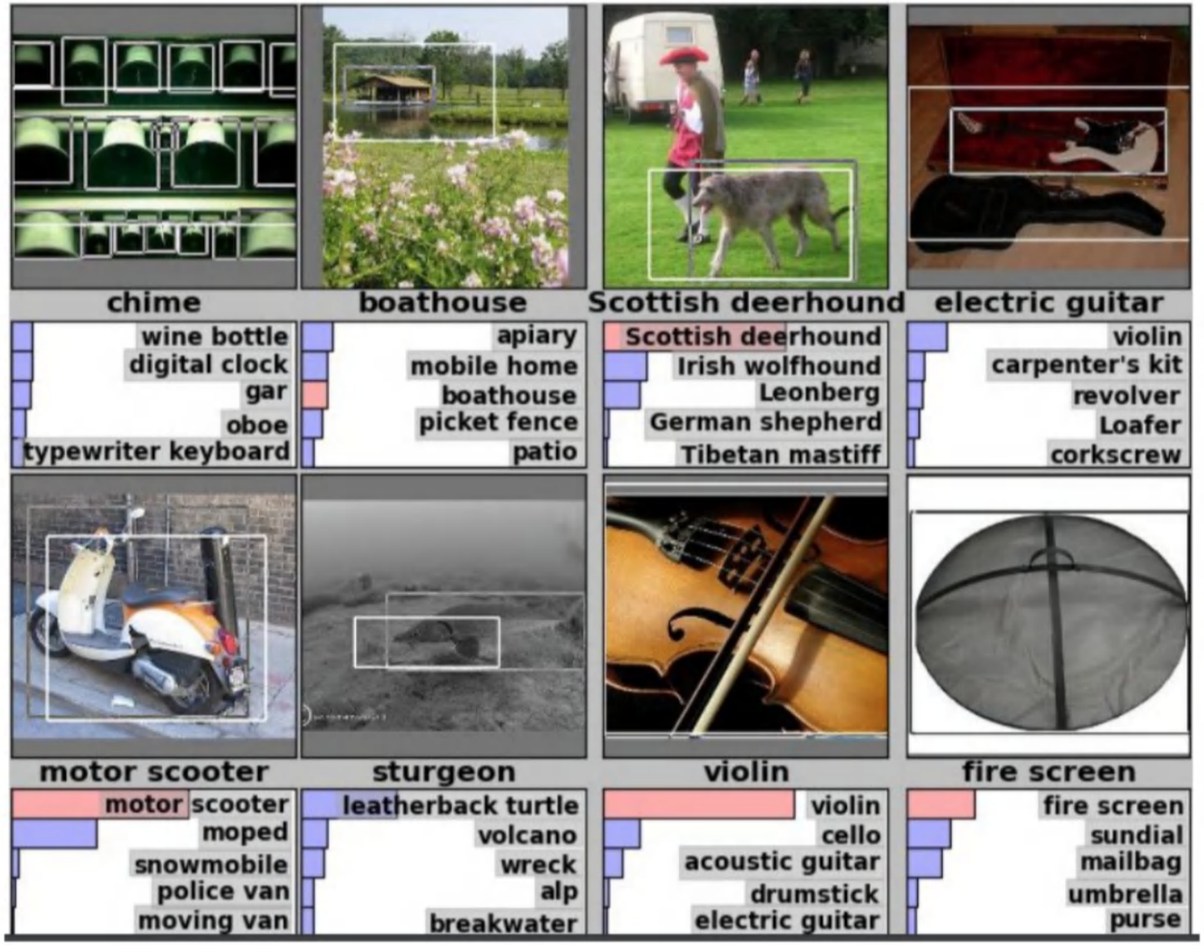
- **Input:** Image
- **Output:** 物体边界框 (x, y, w, h)
- **Evaluation metric:** 交并准则

→ (x, y, w, h)

Classification + Localization: 识别主体+定位

ImageNet

- 实际上有 识别+定位 2个任务



思路1：看作回归问题

- 4个数字，用L2 loss/欧氏距离损失

Input: image

Neural Net

Output:
Box coordinates
(4 numbers)

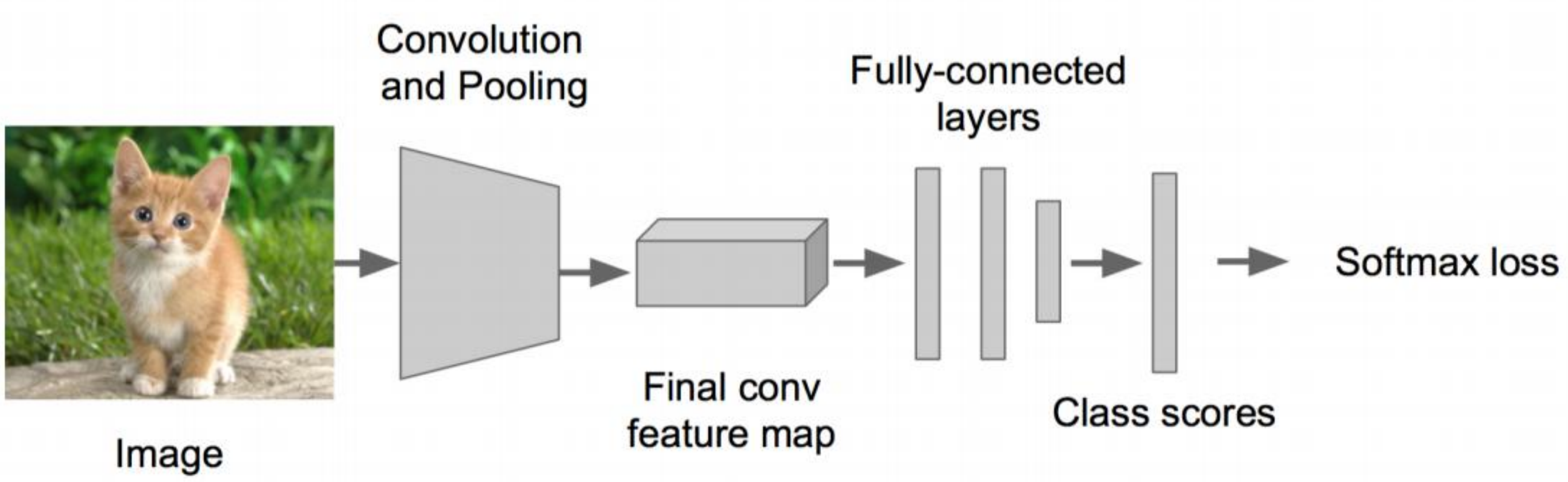
Correct output:
box coordinates
(4 numbers)

Loss:
L2 distance

Only one object,
simpler than detection

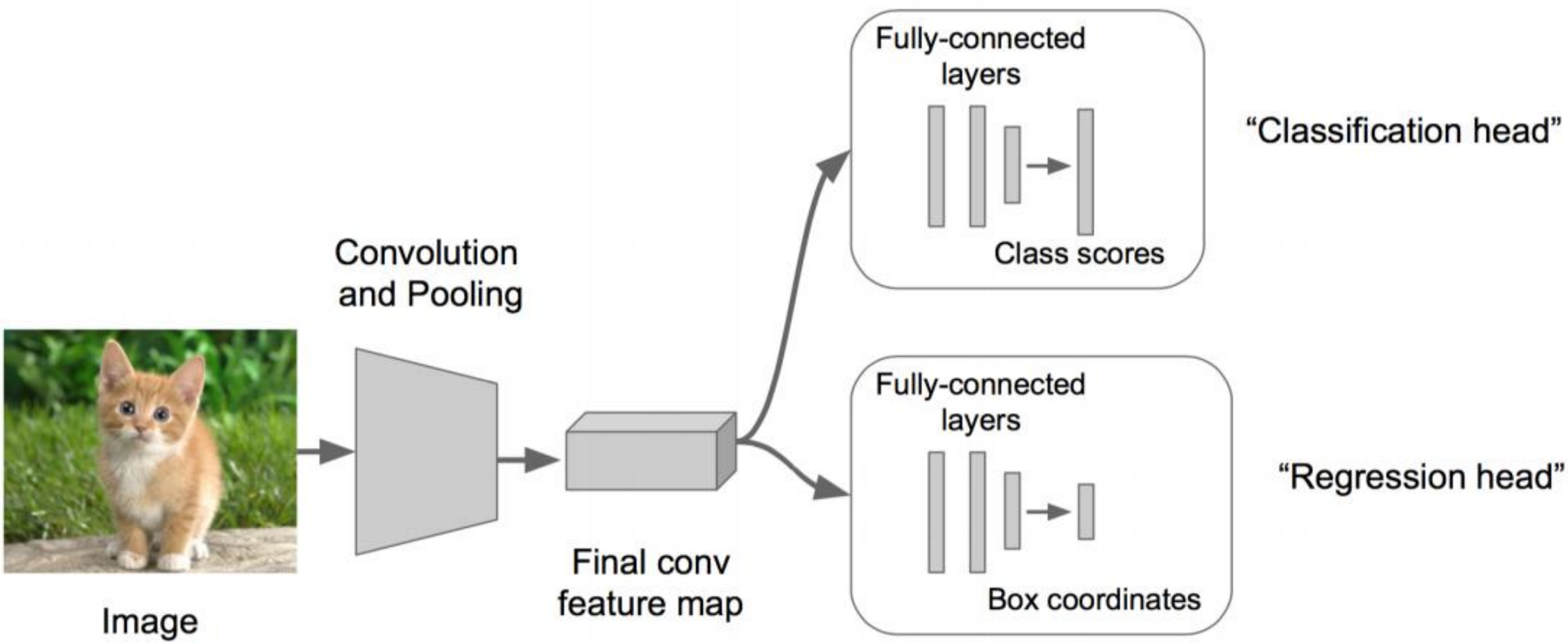
思路1：看作回归问题

- 步骤1:
  - 先解决简单问题，搭一个识别图像的神经网络
  - 在AlexNet VGG GoogleLenet ResNet上fine-tune一下



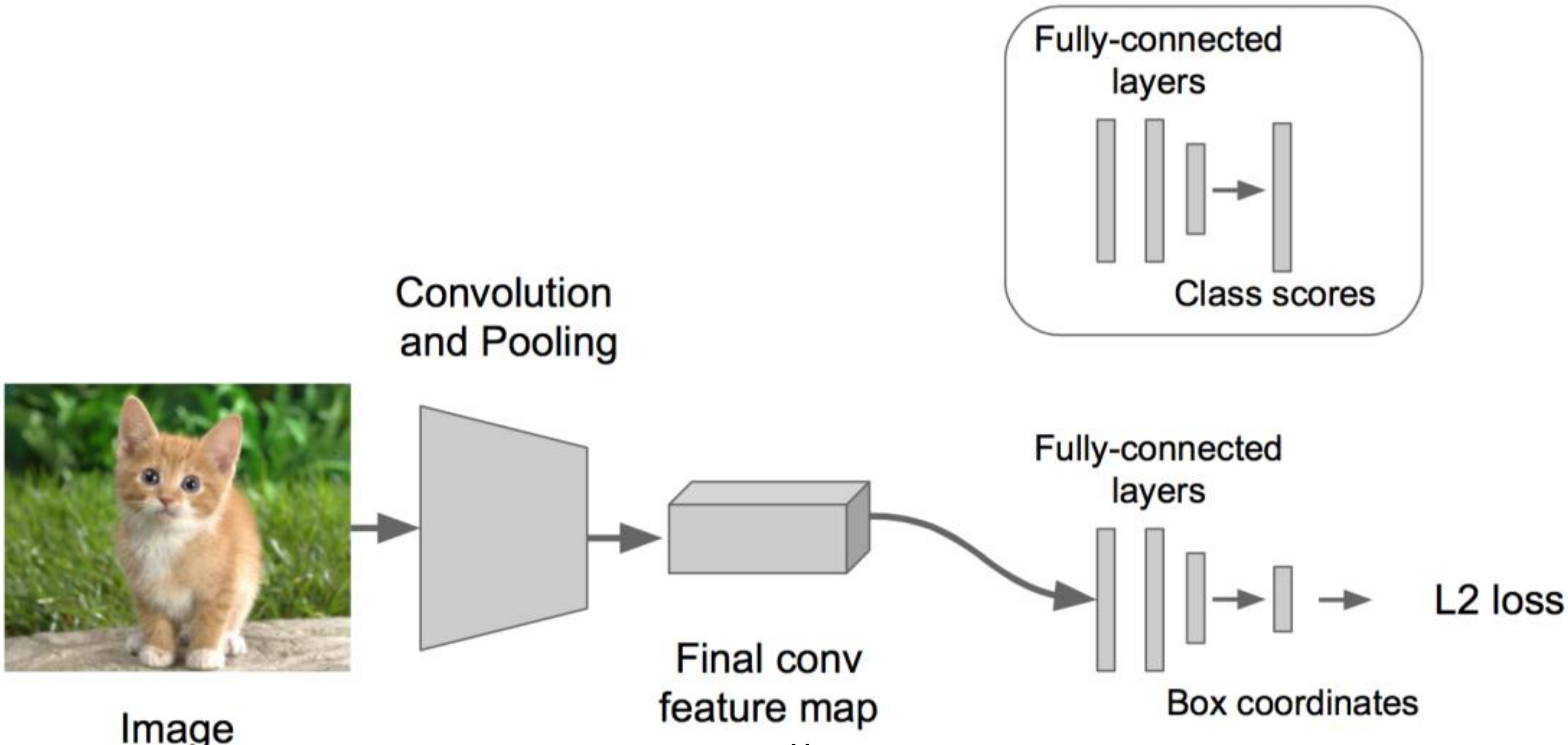
思路1：看作回归问题

- 步骤2:
  - 在上述神经网络的尾部展开
  - 成为classification + regression模式



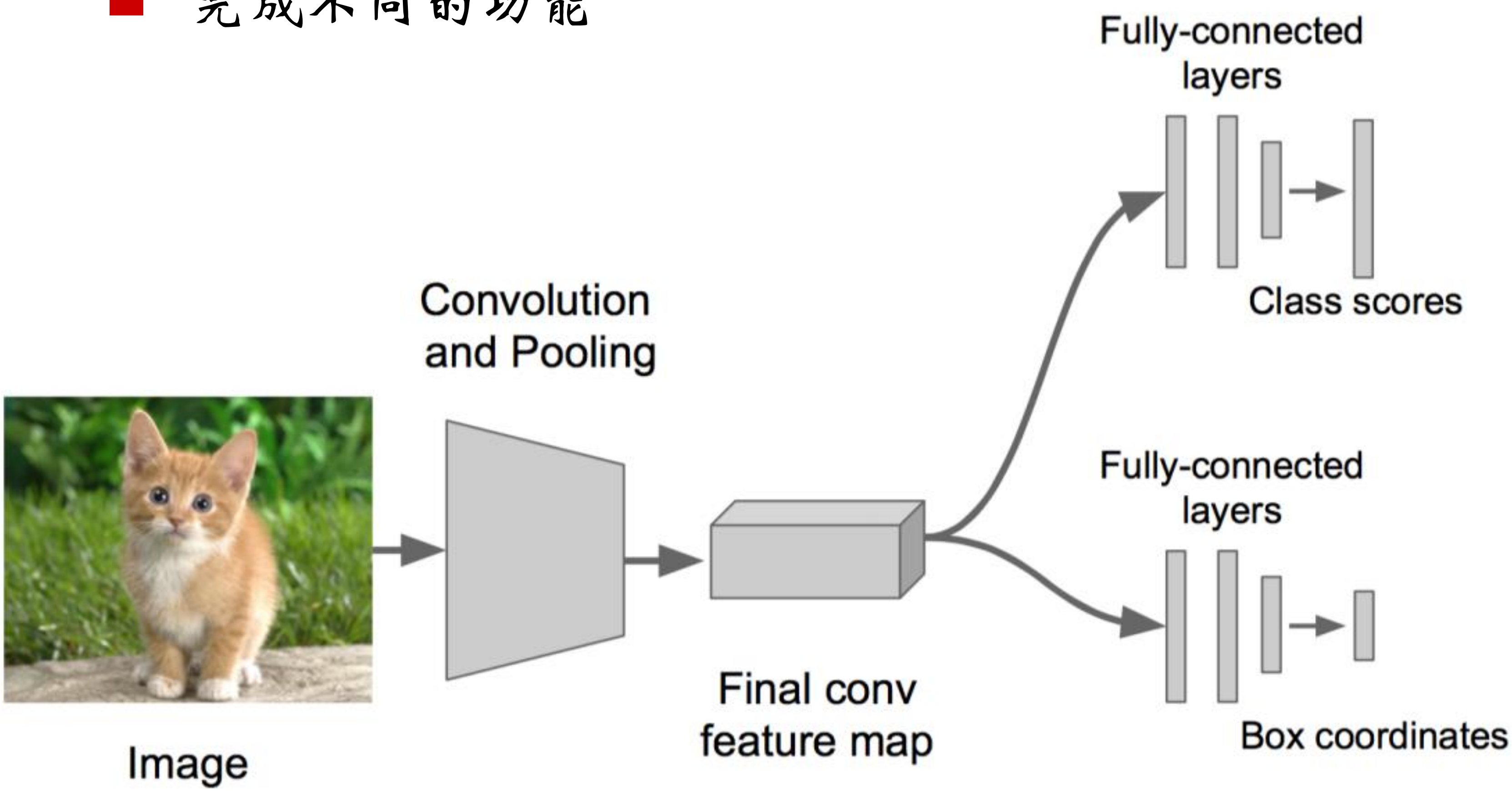
思路1：看作回归问题

- 步骤3:
  - Regression(回归)部分用欧氏距离损失
  - 使用SGD训练



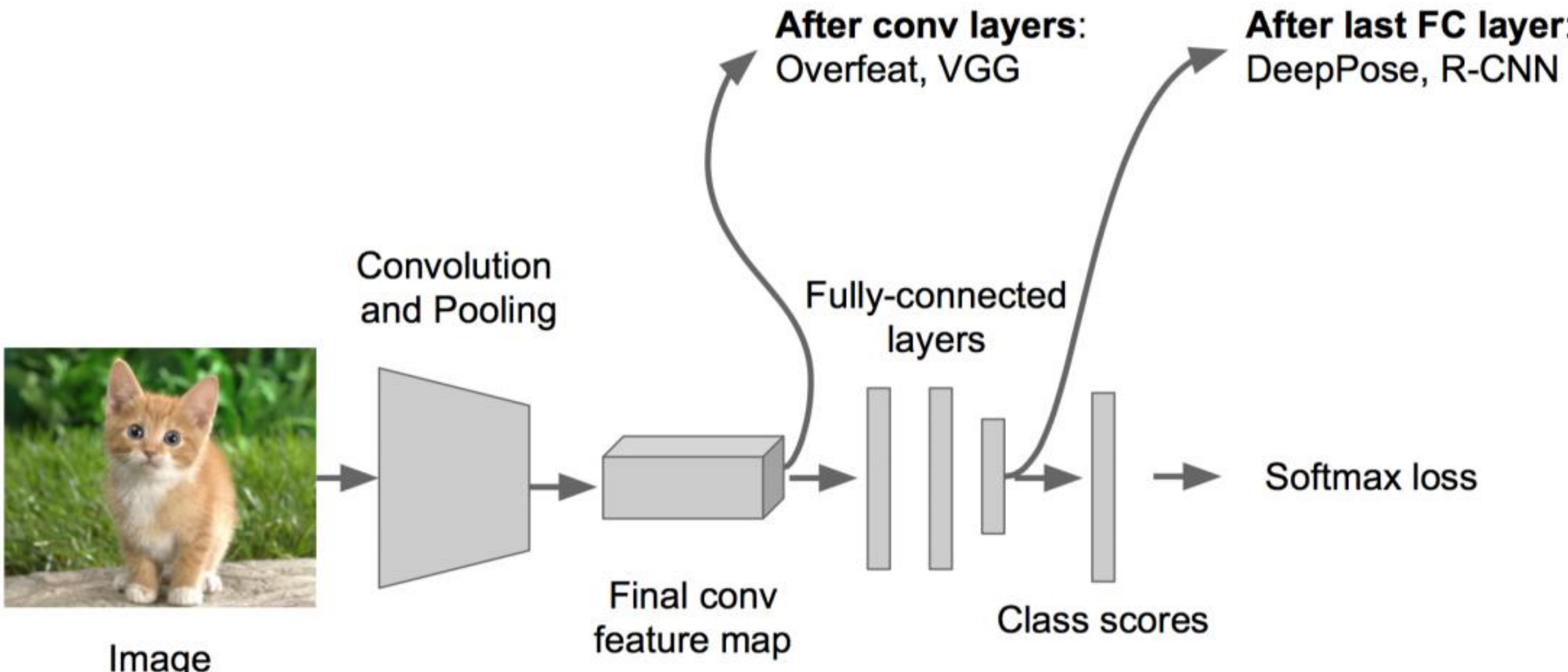
思路1：看作回归问题

- 步骤4:
  - 预测阶段把2个“头部”模块拼上
  - 完成不同的功能



思路1：看作回归问题

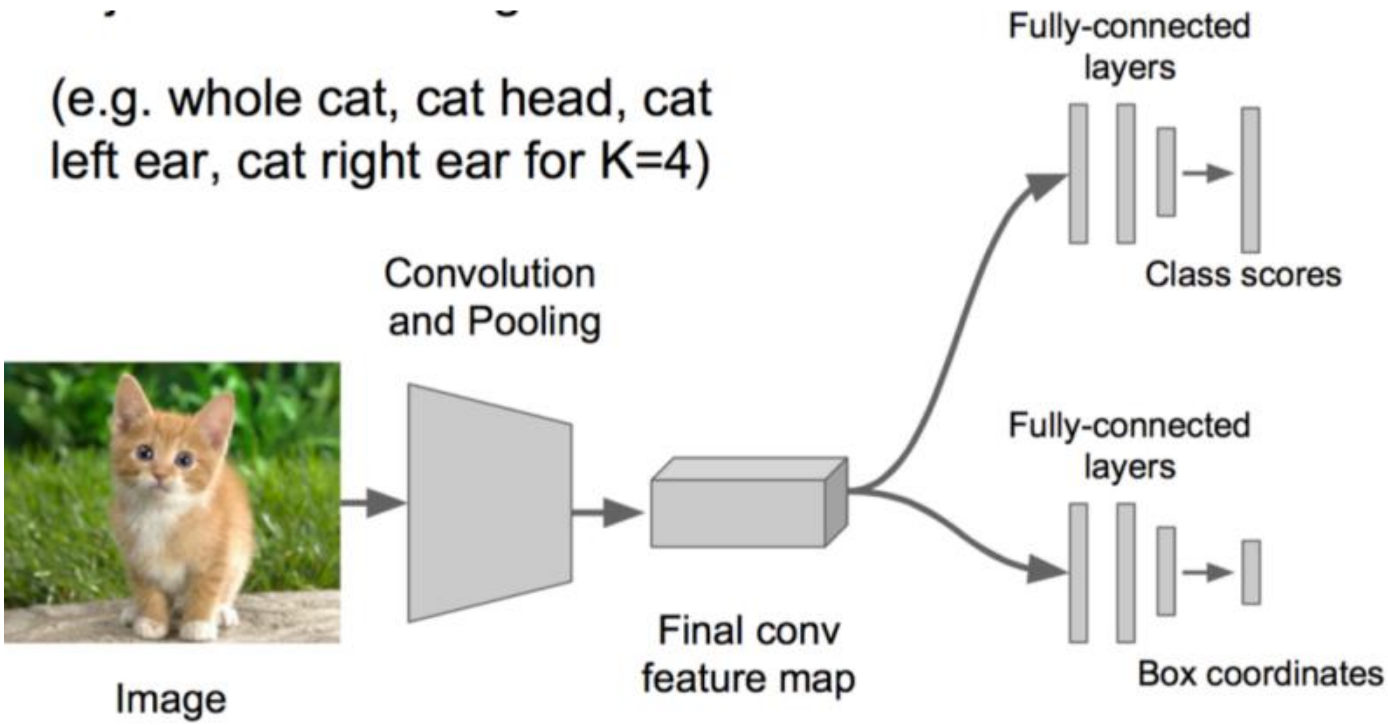
- Regression(回归)的模块部分加在什么位置？
  - ① (最后的)卷积层后
  - ② 全连接层后



思路1：看作回归问题

- 能否对主体有更细致的识别？
  - 提前规定好有K个组成部分
  - 做成K个部分的回归

(e.g. whole cat, cat head, cat left ear, cat right ear for K=4)



K x 4 numbers
(one box per object)

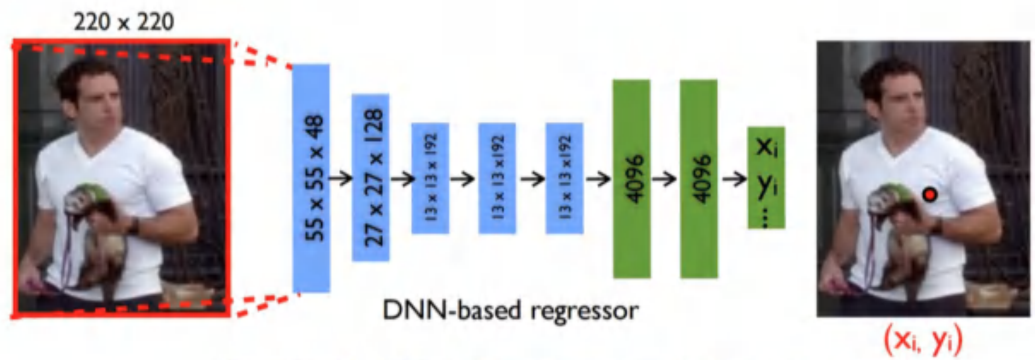
思路1：看作回归问题

- 应用：如何识别人的姿势？
  - 每个人的组成部分是固定的
  - 对K个组成部分(关节)做回归预测 => 首尾相接的线段

Represent a person by K joints

Regress (x, y) for each joint from last fully-connected layer of AlexNet

(Details: Normalized coordinates, iterative refinement)

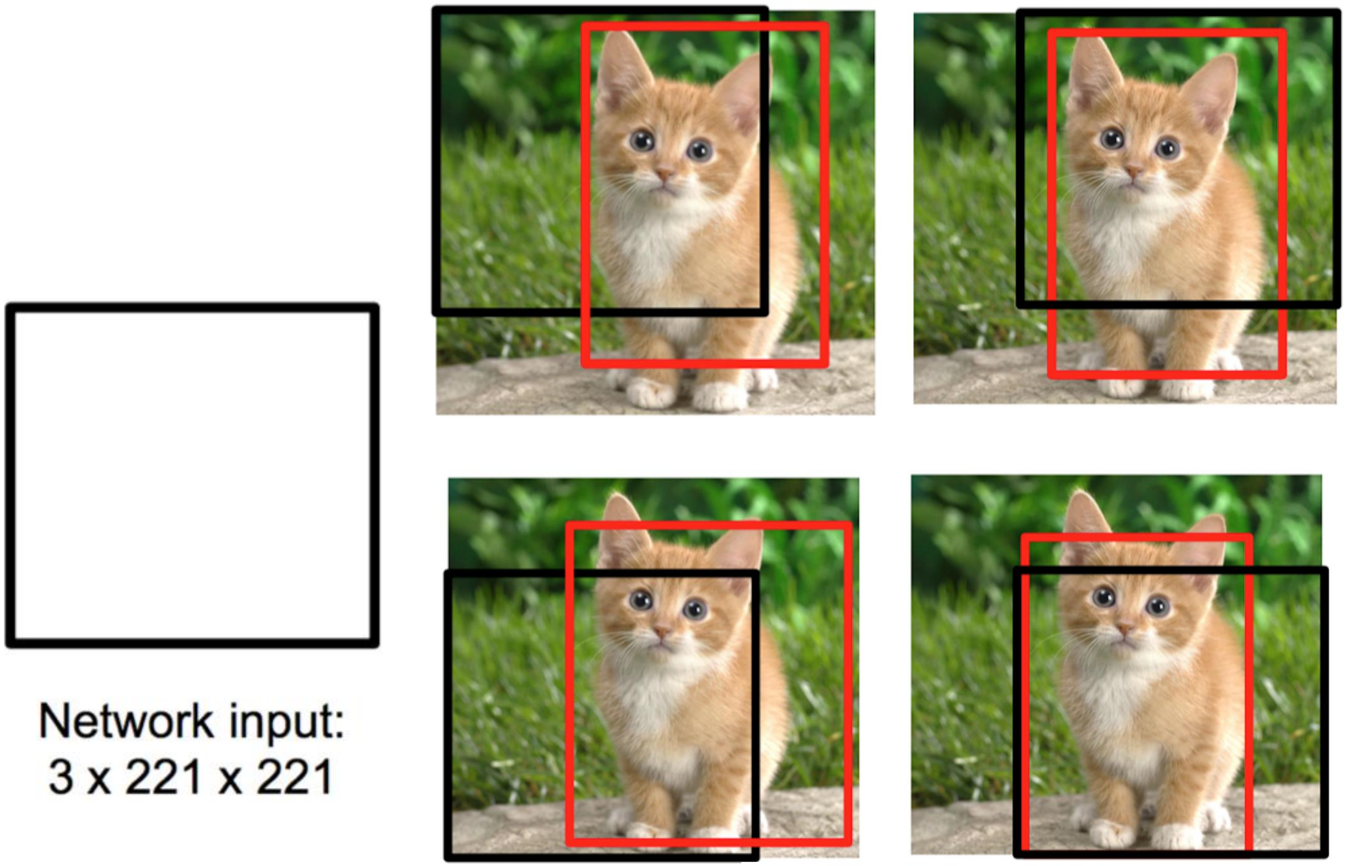


Toshev and Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks", CVPR 2014

思路2：图窗+识别与整合

- 类似刚才的classification + regression思路
- 咱们取不同的大小的 “框”
- 让框出现在不同的位置
- 判定得分
- 按照得分高低对“结果框”做抽取和合并

思路2：图窗+识别与整合



Network input:
3 x 221 x 221

Larger image:
3 x 257 x 257

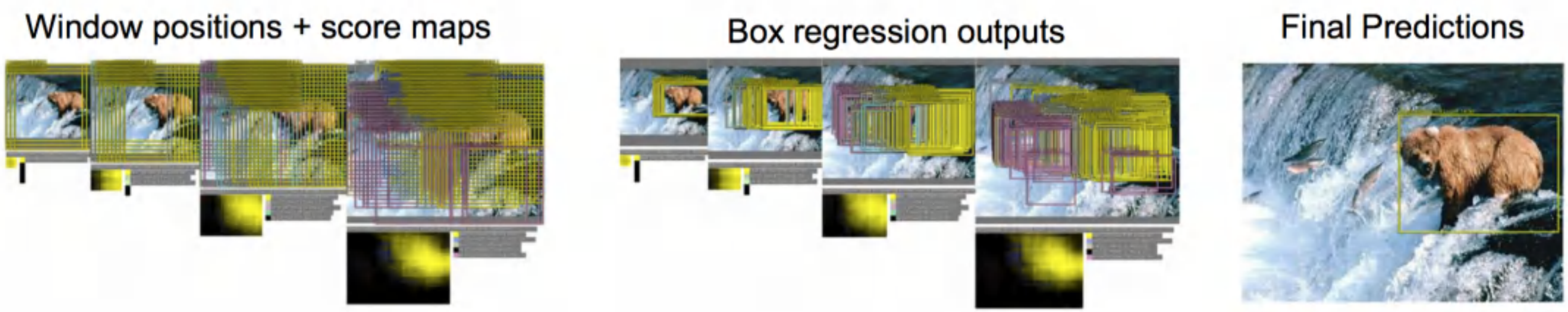
0.5	0.75
0.6	0.8

Classification scores:
P(cat)

思路2：图窗+识别与整合

- 实际应用时
  - 尝试各种大小窗口
  - 甚至会在窗口上再做一些“回归”的事情

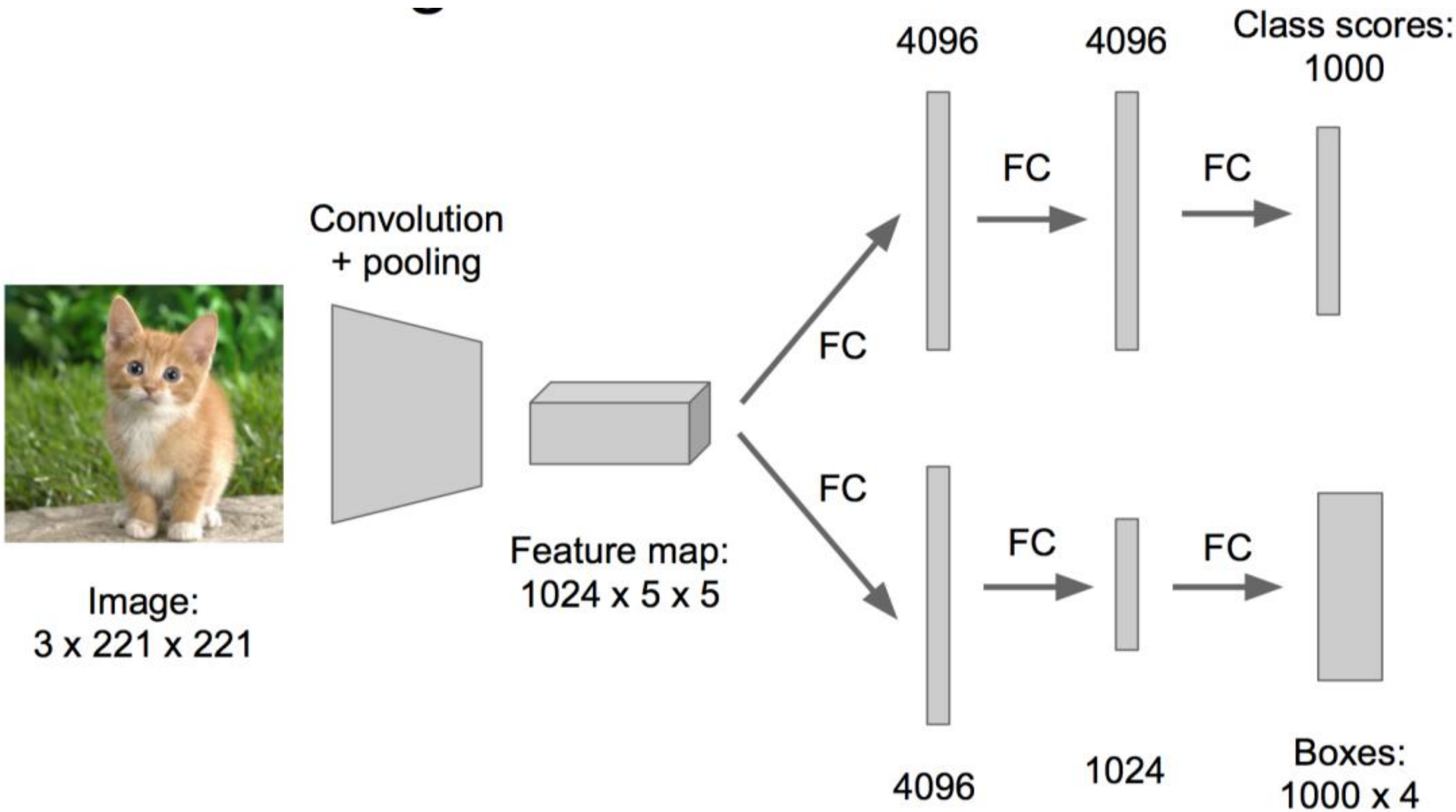
In practice use many sliding window locations and multiple scales



Sermanet et al, “Integrated Recognition, Localization and Detection using Convolutional Networks”, ICLR 2014

思路2：图窗+识别与整合

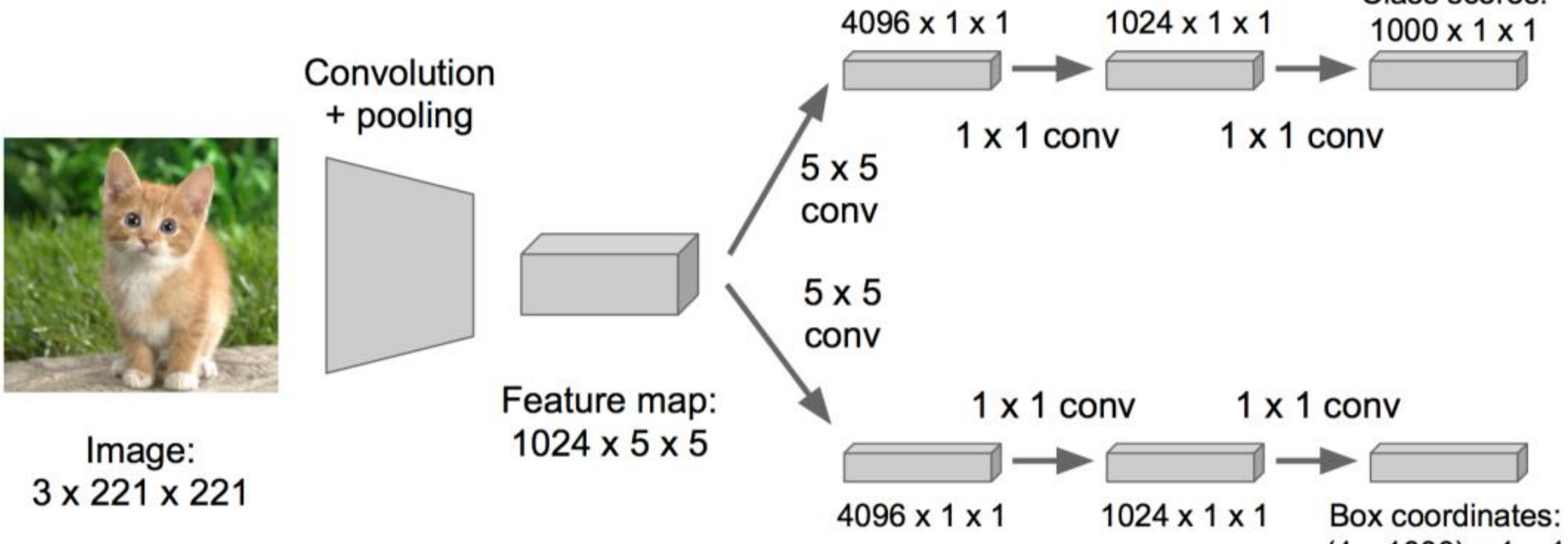
- 想办法克服一下过程中的“参数多”与“计算慢”
  - 最初的形式如下



思路2：图窗+识别与整合

- 想办法克服一下过程中的“参数多”与“计算慢”
  - 用多卷积核的卷积层 替换 全连接层
  - 降低参数量

Efficient sliding window by converting fully-connected layers into convolutions

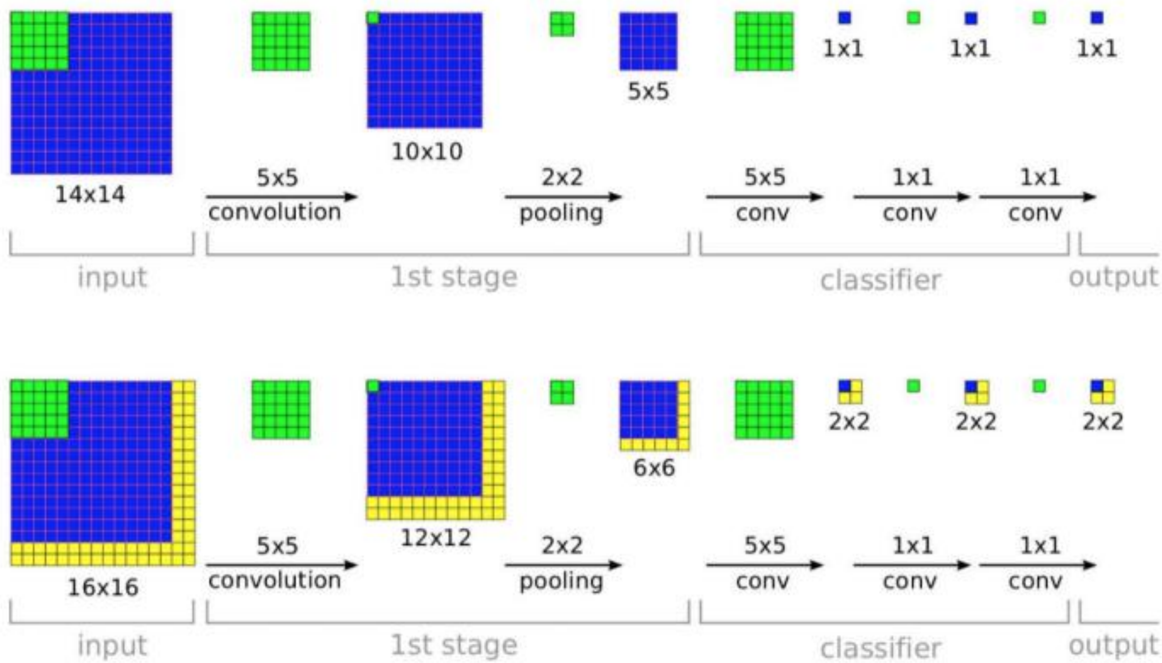


图像识别+定位

- 想办法克服一下过程中的“参数多”与“计算慢”
  - 测试/识别 阶段的计算是可复用的(小卷积)
  - 加速计算

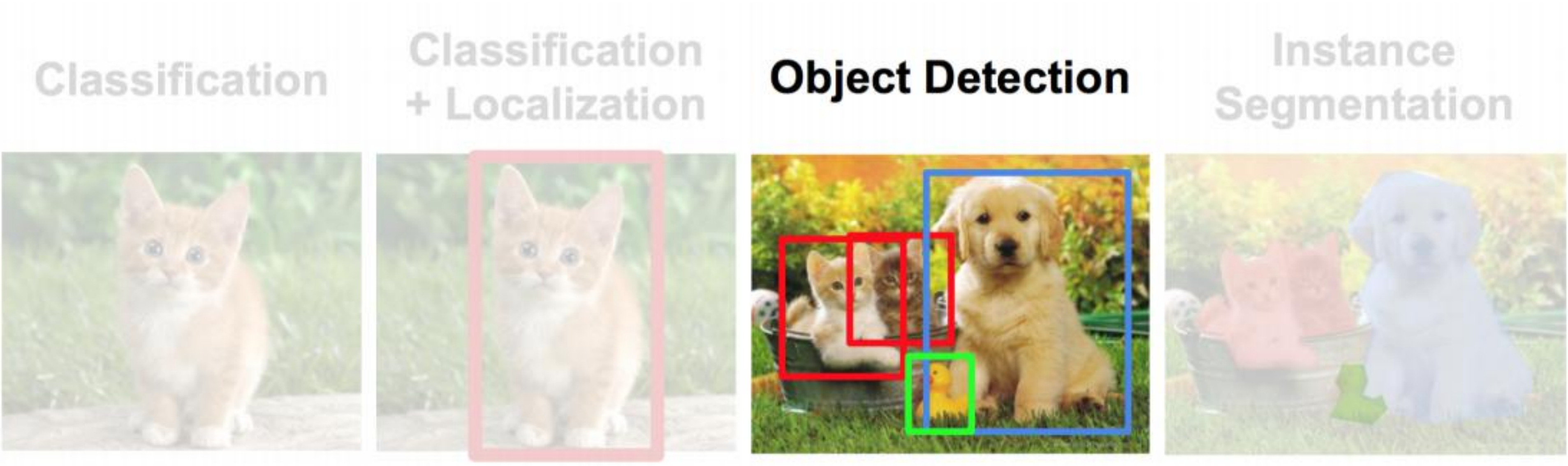
Training time: Small image, 1 x 1 classifier output

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



Sermanet et al, “Integrated Recognition, Localization and Detection using Convolutional Networks”, ICLR 2014

物体识别


Classification
Classification + Localization
Object Detection
Instance Segmentation

物体识别

- 再次看做回归问题？

DOG, (x, y, w, h)
CAT, (x, y, w, h)
CAT, (x, y, w, h)
DUCK (x, y, w, h)

= 16 numbers

物体识别

- 其实你不知道图上有多少个物体...

CAT, (x, y, w, h)
CAT, (x, y, w, h)
....
CAT (x, y, w, h)

物体识别

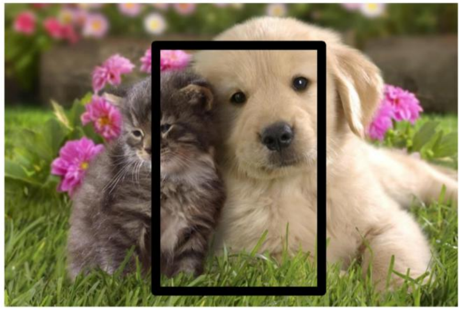
- 试着看做分类问题？

CAT? NO

DOG? NO

CAT? YES!

DOG? NO



CAT? NO

DOG? NO

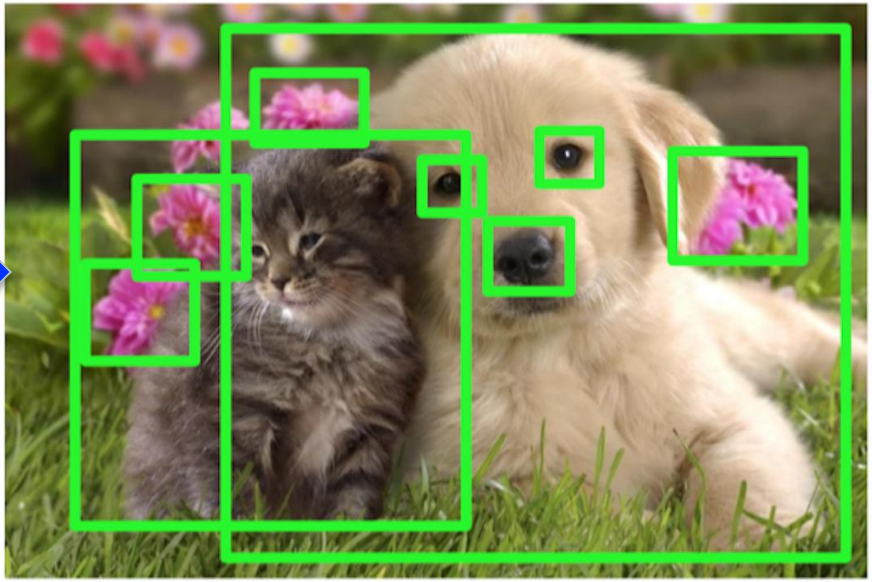
物体识别

- 看做分类问题，难点是？
  - 你需要找“很多位置”，给“很多不同大小的框”
  - 你还需要对框内的图像分类(累计很多次)
  - 框的大小不一定对
  - ...

  - 当然，如果你的GPU很强大，恩，那加油做吧...

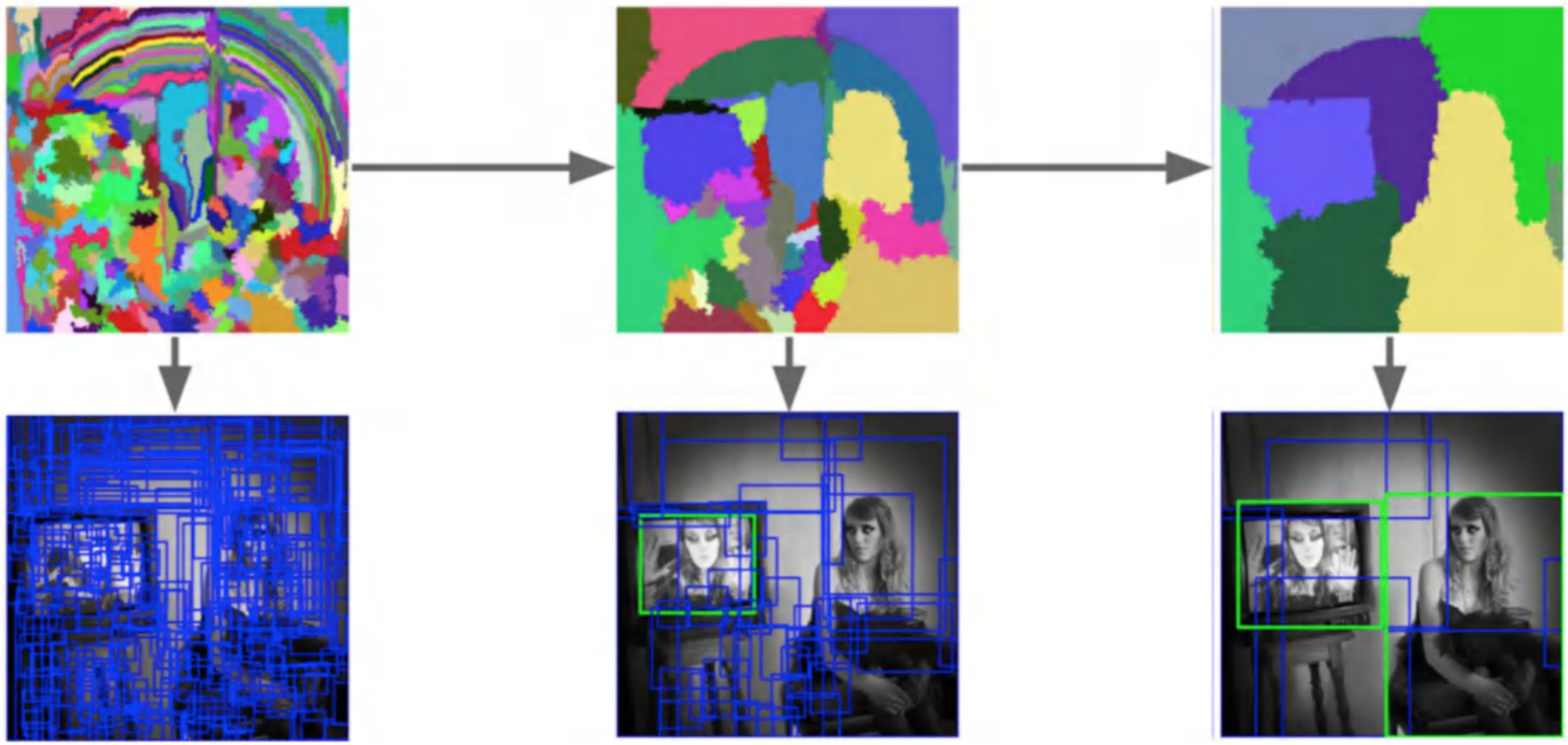
物体识别：边缘策略

- 看做分类问题，有没有办法优化下？
  - 为什么要先给定“框”，能不能找到“候选框”？
  - 想办法先找到“可能包含内容的图框”



物体识别：选择性搜索

- 关于“候选图框”识别，有什么办法？
  - 自下而上融合成“区域”
  - 将“区域”扩充为“图框”



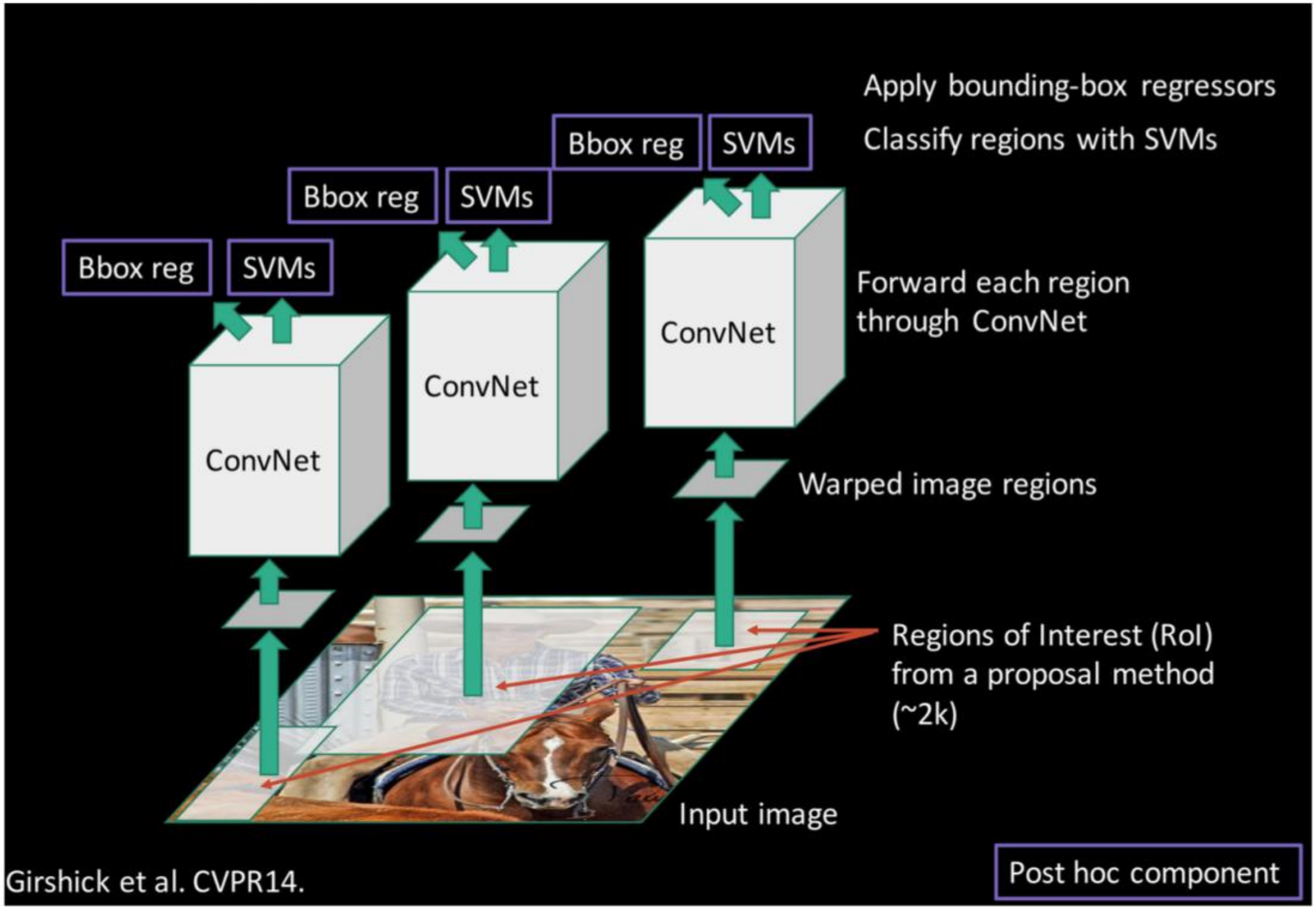
Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013

“图框”候选：其他方式？

Method	Approach	Outputs Segments	Outputs Score	Control #proposals	Time (sec.)	Repea-tability	Recall Results	Detection Results
Bing [18]	Window scoring		✓	✓	0.2	★★★	★	·
CPMC [19]	Grouping	✓	✓	✓	250	-	★★	★
EdgeBoxes [20]	Window scoring		✓	✓	0.3	★★	★★★	★★★
Endres [21]	Grouping	✓	✓	✓	100	-	★★★	★★
Geodesic [22]	Grouping	✓		✓	1	★	★★★	★★
MCG [23]	Grouping	✓	✓	✓	30	★	★★★	★★★
Objectness [24]	Window scoring		✓	✓	3	·	★	·
Rahtu [25]	Window scoring		✓	✓	3	·	·	★
RandomizedPrim's [26]	Grouping	✓		✓	1	★	★	★★
Rantalankila [27]	Grouping	✓		✓	10	★★	·	★★
Rigor [28]	Grouping	✓		✓	10	★	★★	★★
SelectiveSearch [29]	Grouping	✓	✓	✓	10	★★	★★★	★★★
Gaussian				✓	0	·	·	★
SlidingWindow				✓	0	★★★	·	·
Superpixels		✓			1	★	·	·
Uniform				✓	0	·	·	·

Hosang et al, “What makes for effective detection proposals?”, PAMI 2015

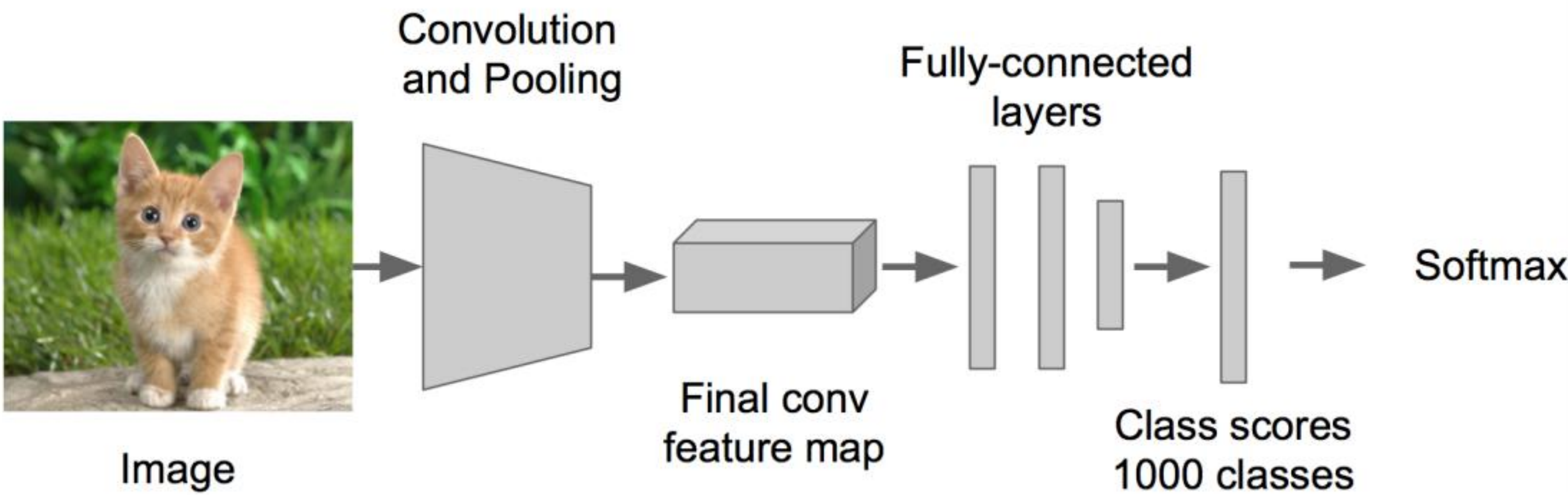
R-CNN



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

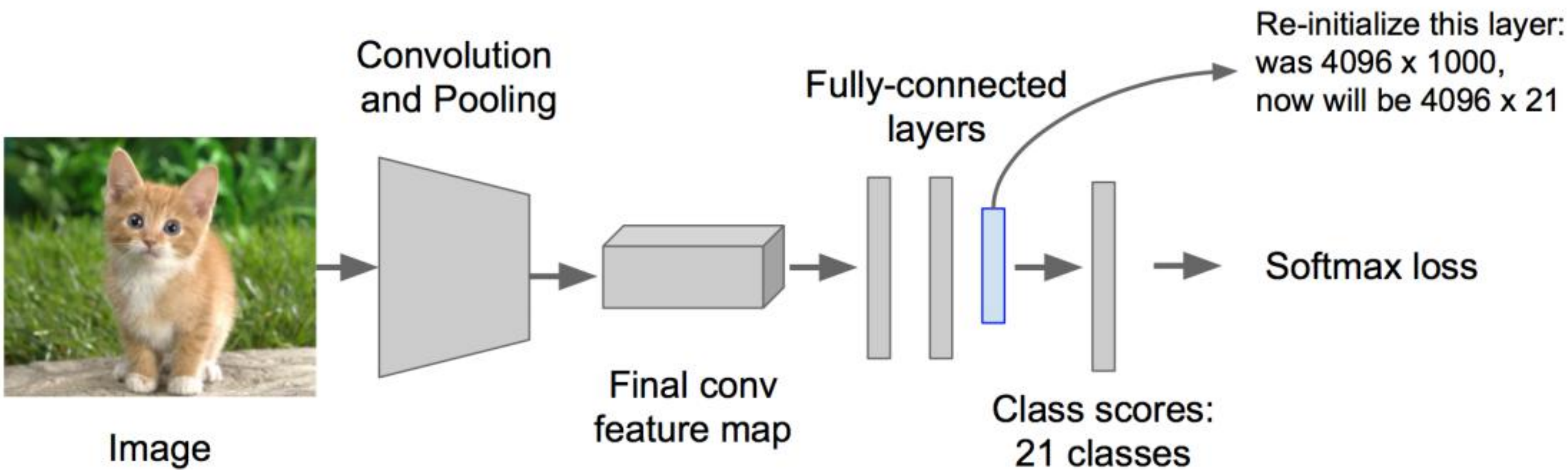
R-CNN

- 步骤1：找一个预训练好的模型(Alexnet,VGG)针对你的场景做fine-tune



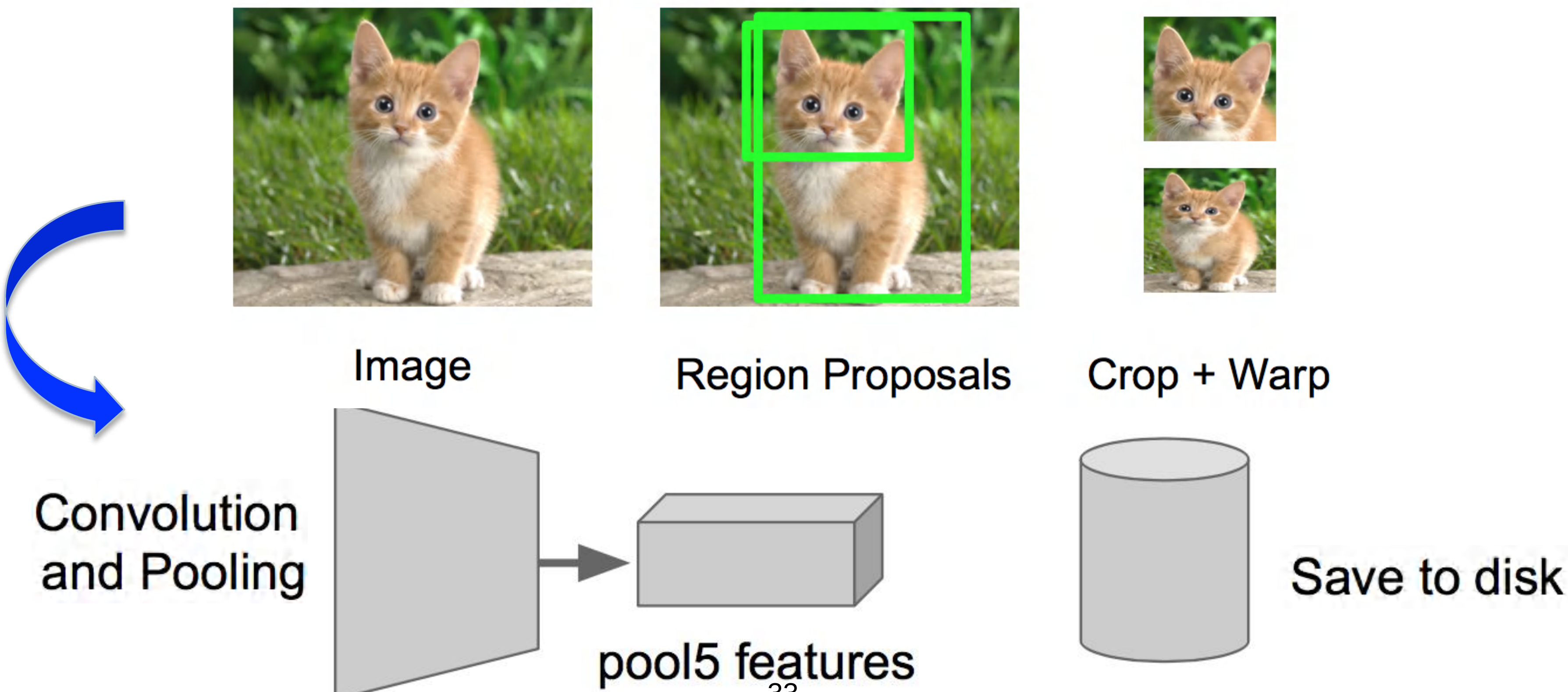
R-CNN

- 步骤2：fine-tuning模型
  - 比如20个物体类别+1个背景



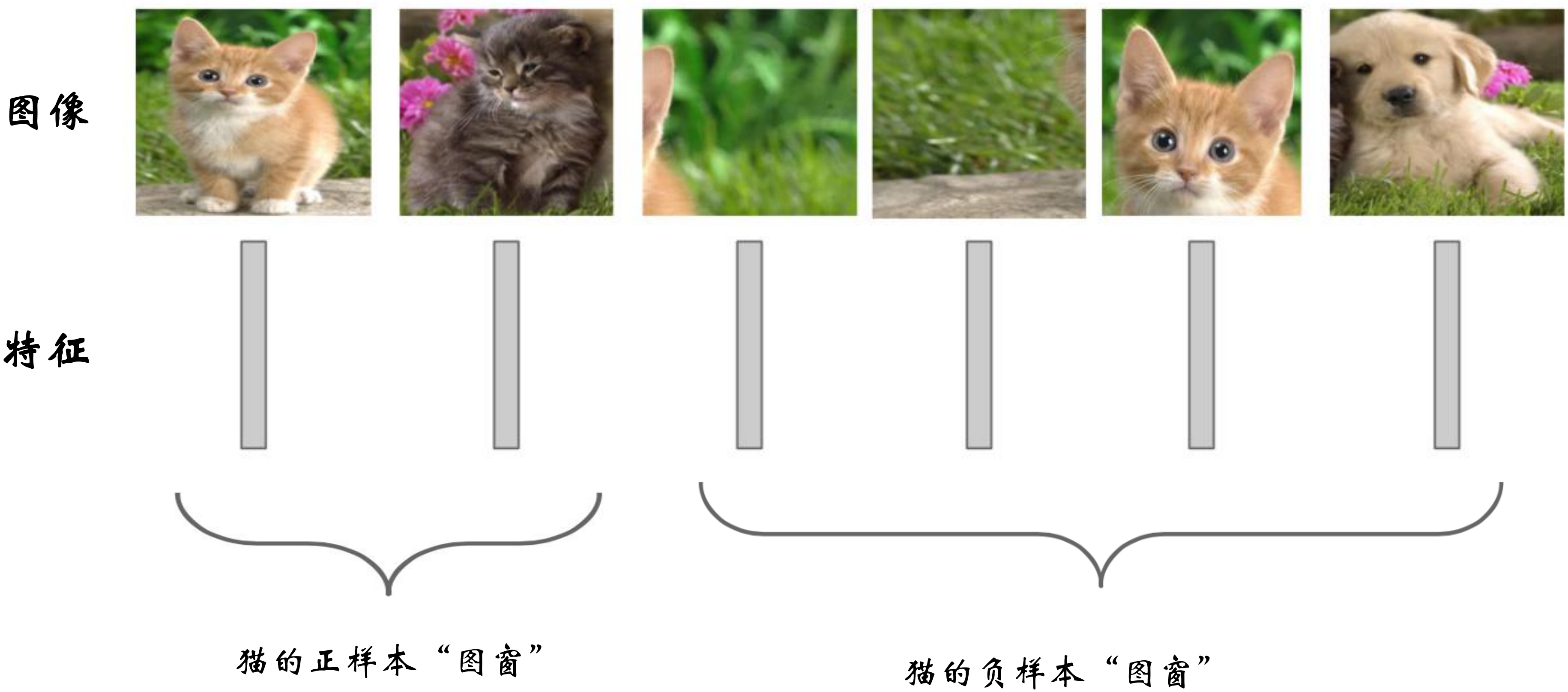
R-CNN

- 步骤3：抽取图片特征
  - 用“图框候选算法”抠出图窗
  - Resize后用CNN做前向运算，取第5个池化层做特征
  - 存储抽取的特征到硬盘/数据库上



R-CNN

- 步骤4：训练SVM识别是某个物体或者不是(2分类)

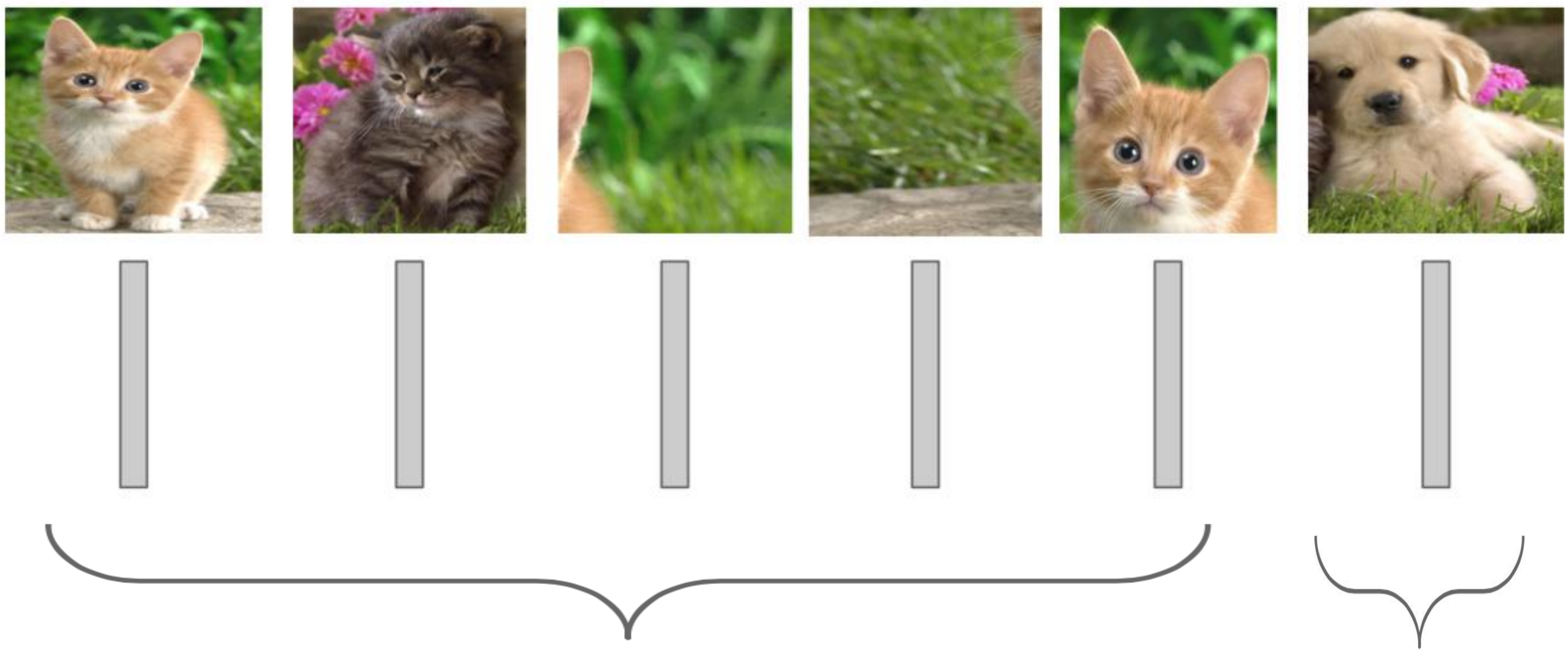


R-CNN

- 步骤4：训练SVM识别是某个物体或者不是(2分类)

图像

特征



狗的负样本“图窗”

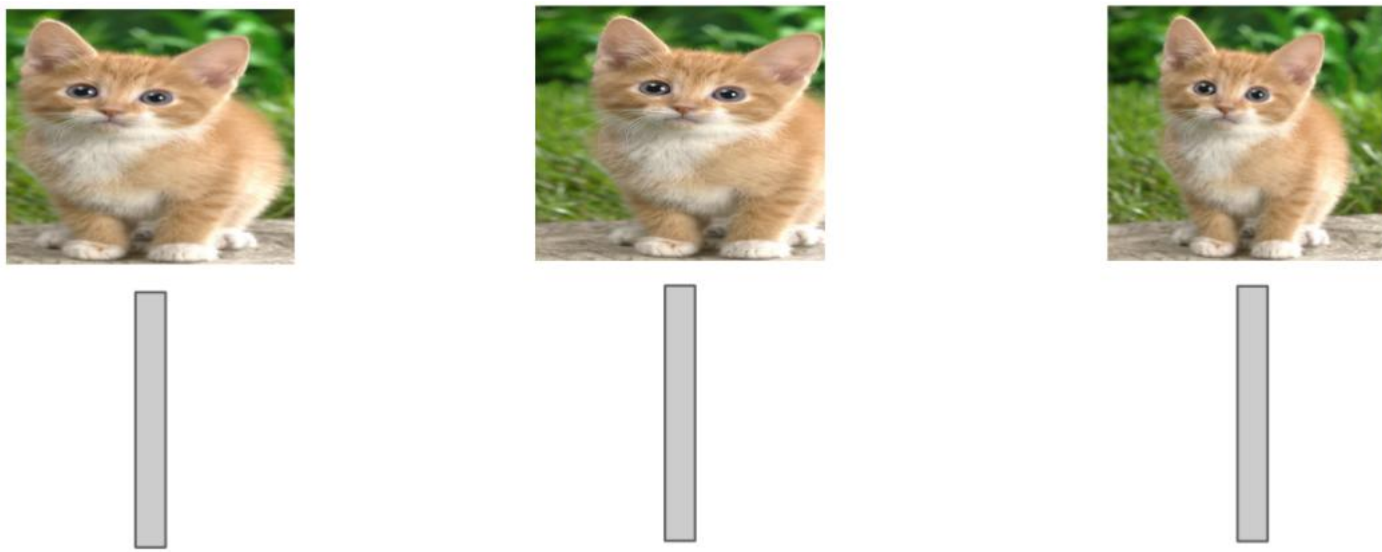
狗的正样本“图窗”

R-CNN

- 步骤5：bbox regression
  - 微调图窗区域

图像

特征

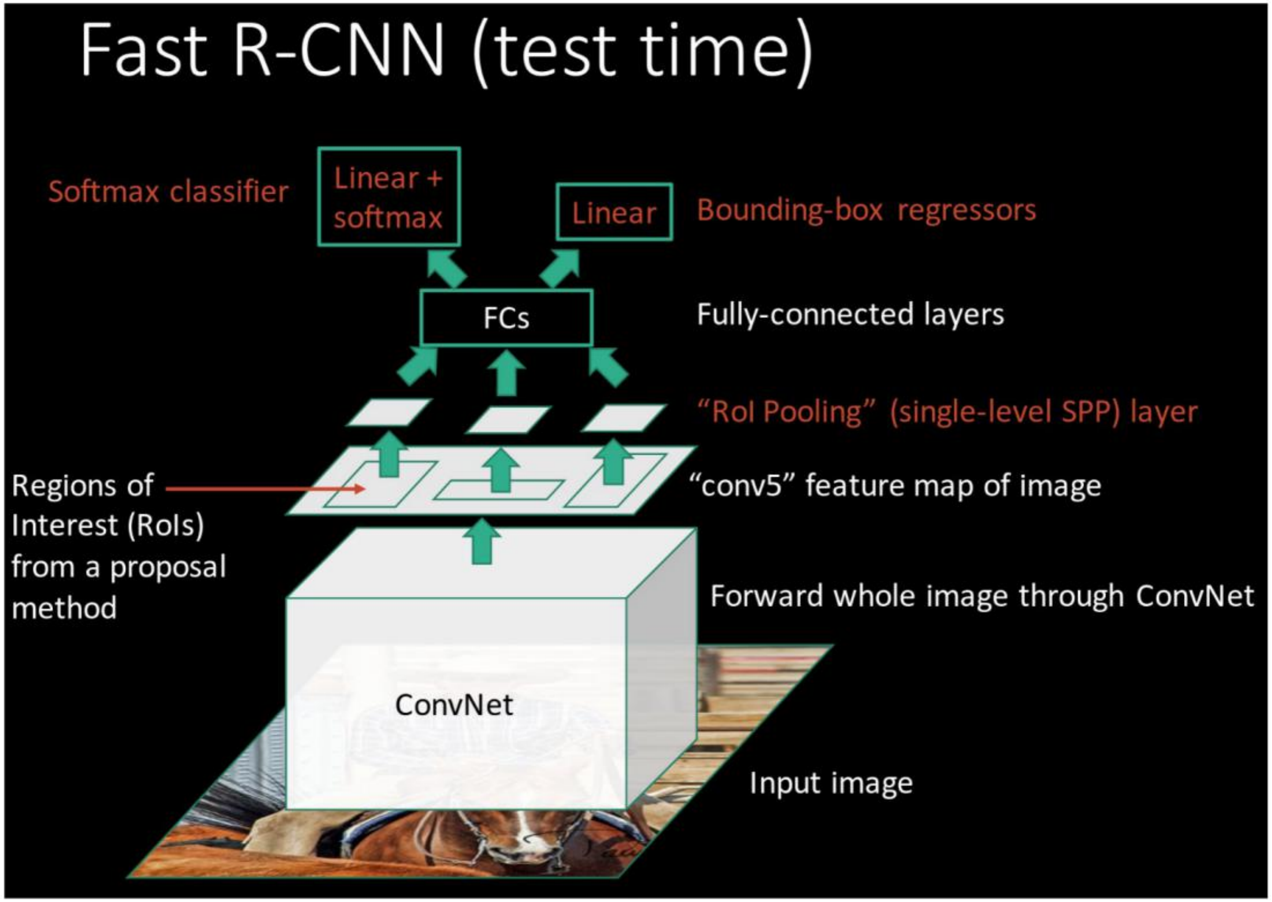


回归得到 dx, dy, dw, dh 位置调整

(0, 0, 0, 0)	(.25, 0, 0, 0)	(0, 0, -0.125, 0)
Proposal is good	Proposal too far to left	Proposal too wide

R-CNN => Fast-rcnn

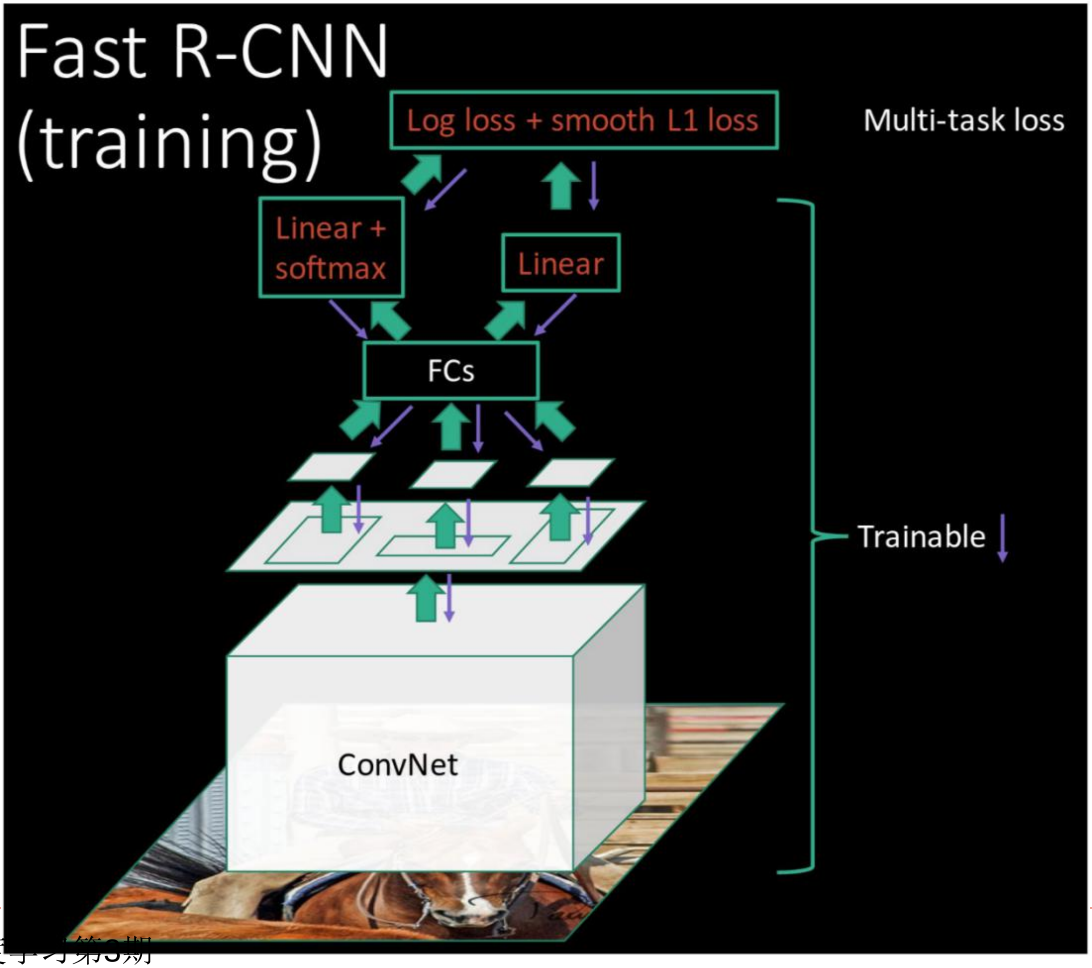
- 针对R-CNN的改进1
  - 共享图窗计算，从而加速



Girschick, "Fast R-CNN", ICCV 2015

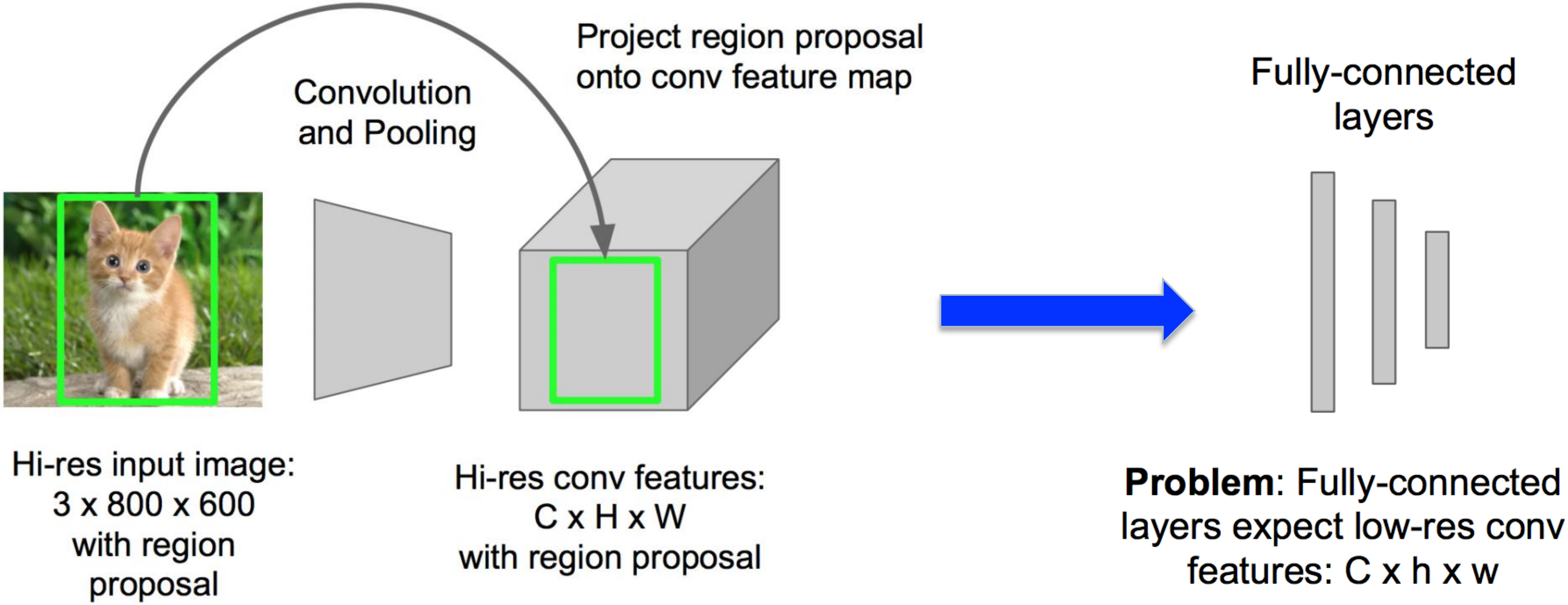
R-CNN => Fast-rcnn

- 针对R-CNN的改进2
  - 直接做成端到端的系统



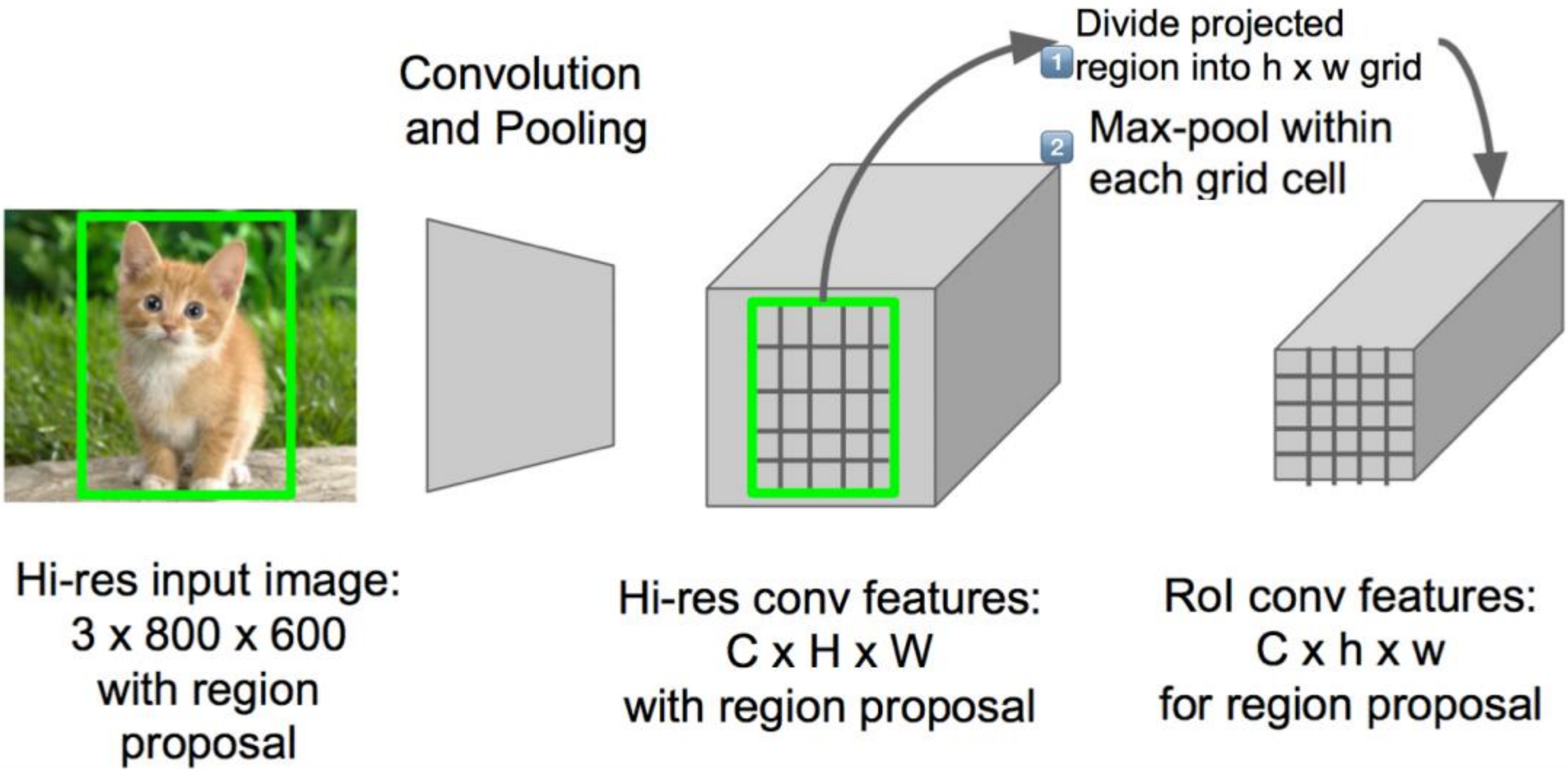
R-CNN => Fast-rcnn

- 关于RIP：Region of Interest Pooling



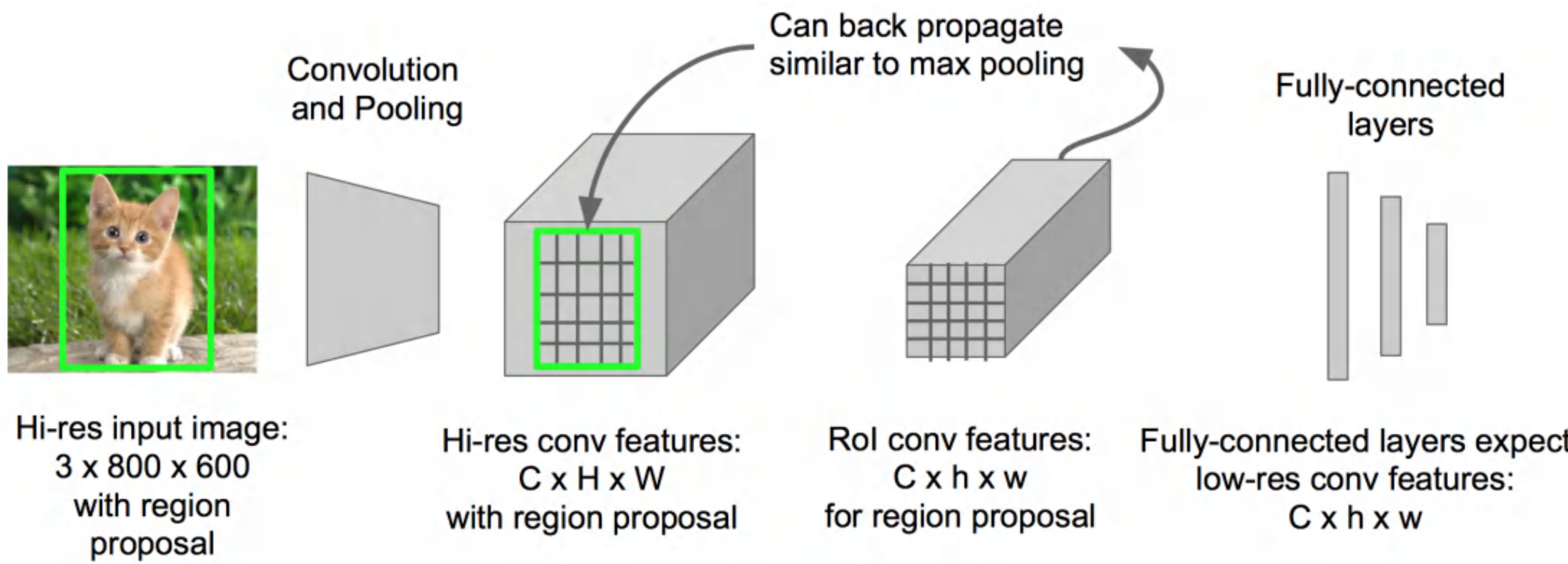
R-CNN => Fast-rcnn

- 维度不匹配怎么办：划分格子grid => 下采样



R-CNN => Fast-rcnn

- RIP：Region of Interest Pooling
  - 映射关系显然是可以还原回去的

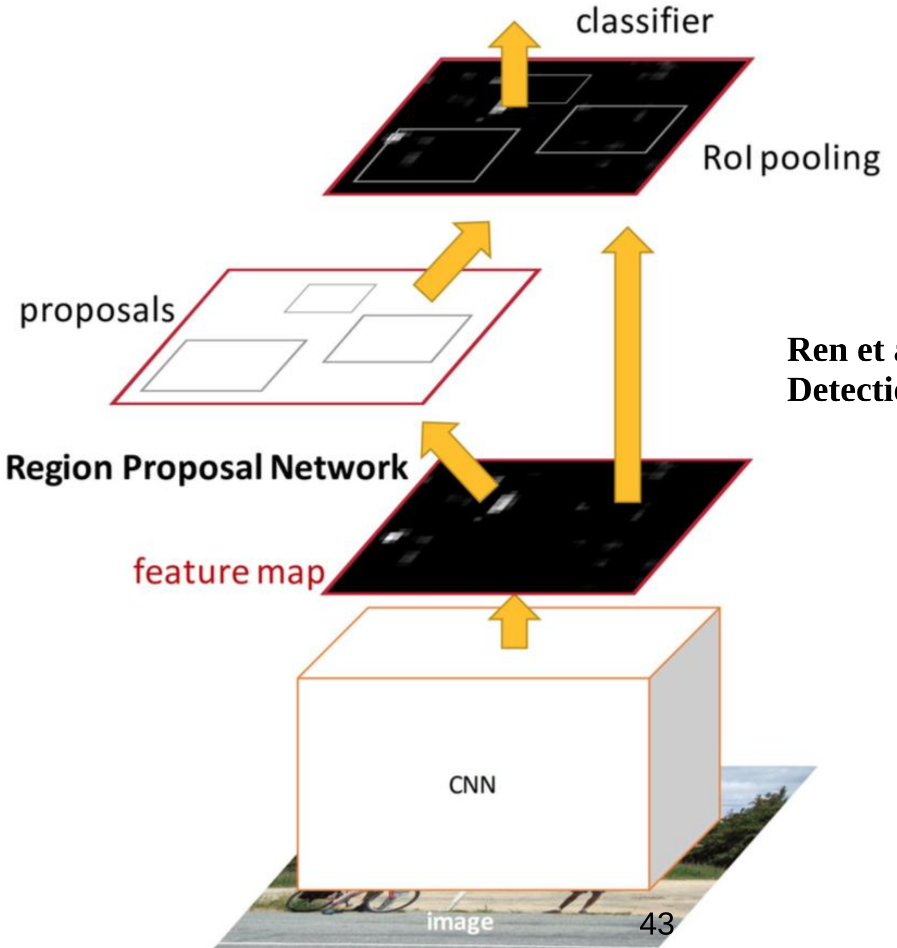


速度对比

		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**

Fast => Faster-rcnn

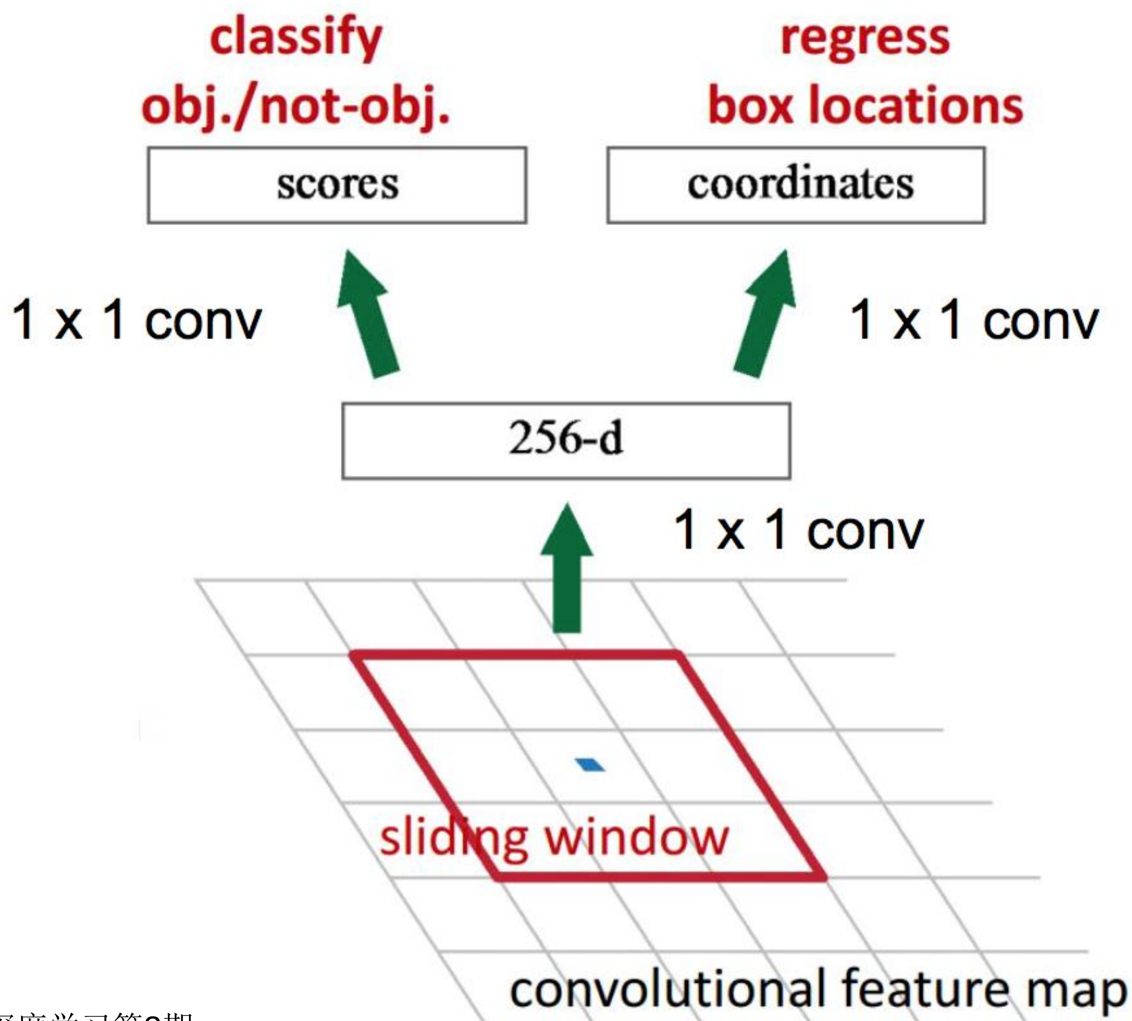
- Region Proposal(候选图窗)一定要另外独立做吗?
  - 一起用RPN做完得了!



Ren et al, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, NIPS 2015

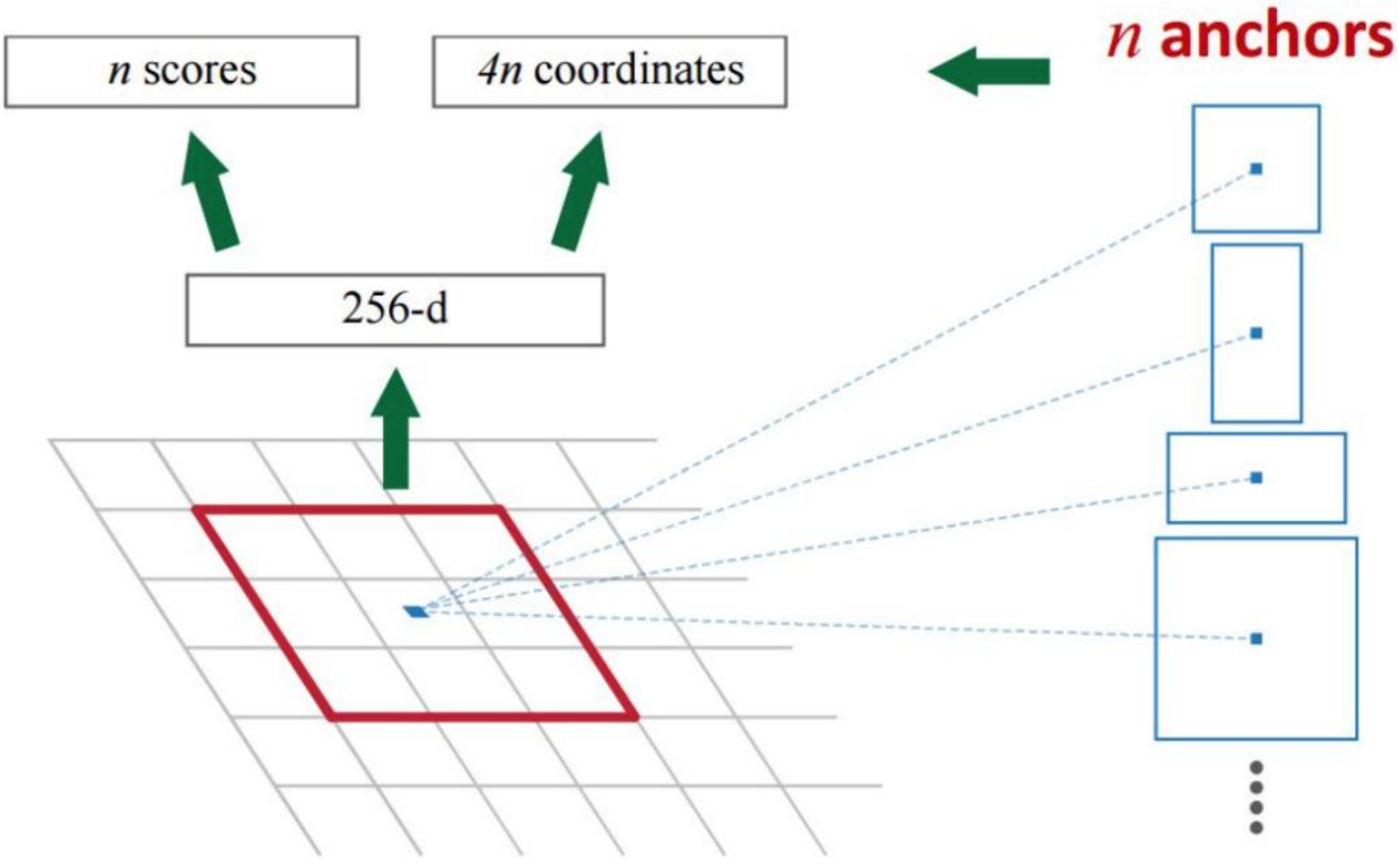
Fast => Faster-rcnn

- 关于RPN：Region Proposal Network



Fast => Faster-rcnn

- 关于RPN：Region Proposal Network



Faster-rcnn

- 关于Faster R-CNN的整个训练过程

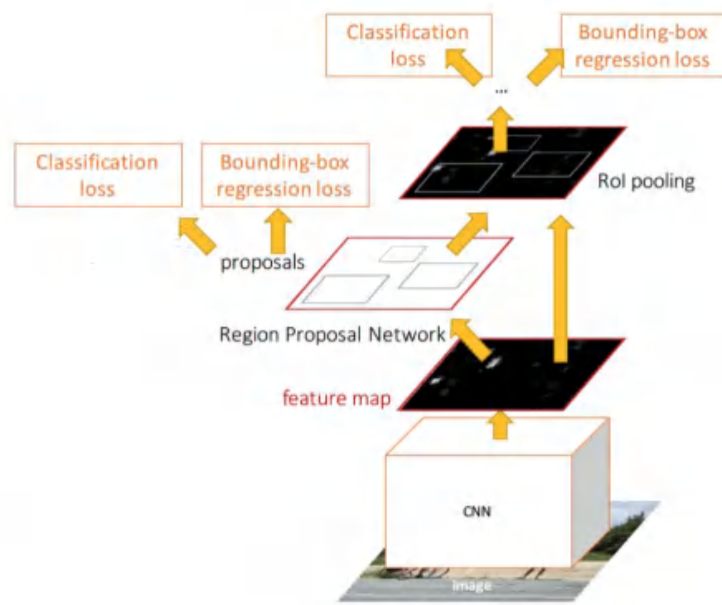
Faster R-CNN: Training

In the paper: Ugly pipeline

- Use alternating optimization to train RPN, then Fast R-CNN with RPN proposals, etc.
- More complex than it has to be

Since publication: Joint training!
One network, four losses

- RPN classification (anchor good / bad)
- RPN regression (anchor -> proposal)
- Fast R-CNN classification (over classes)
- Fast R-CNN regression (proposal -> box)



速度/准度对比

	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**

部分代码与训练数据

R-CNN

(Cafffe + MATLAB): https://github.com/rbgirshick/rcnn （非常慢，参考）

Fast R-CNN

(Caffe + MATLAB): https://github.com/rbgirshick/fast-rcnn（非端到端）

Faster R-CNN

(Caffe + MATLAB): https://github.com/ShaoqingRen/faster_rcnn

(Caffe + Python): https://github.com/rbgirshick/py-faster-rcnn

SSD

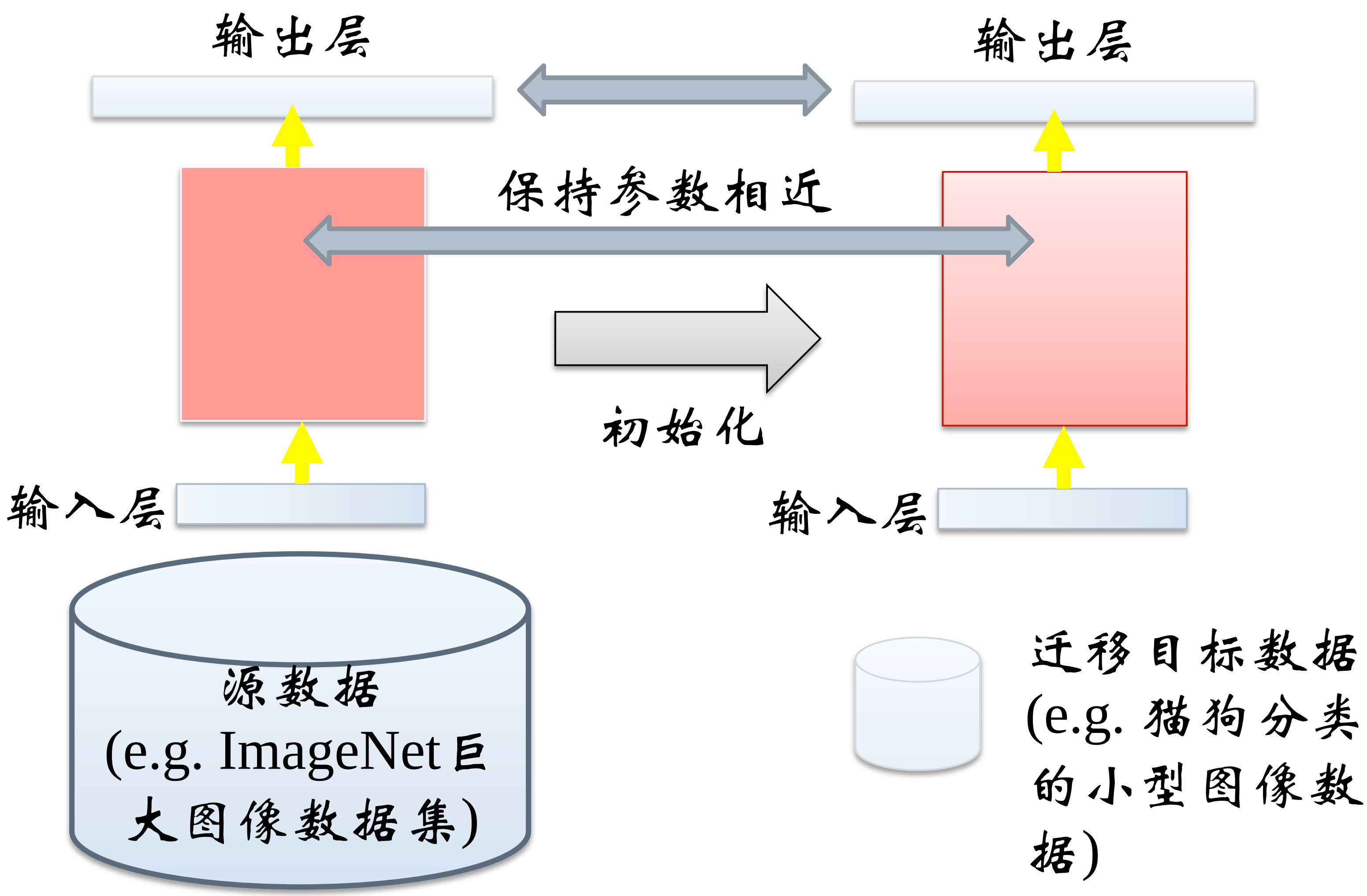
(Caffe + Python)https://github.com/weiliu89/caffe/tree/ssd

R-FCN

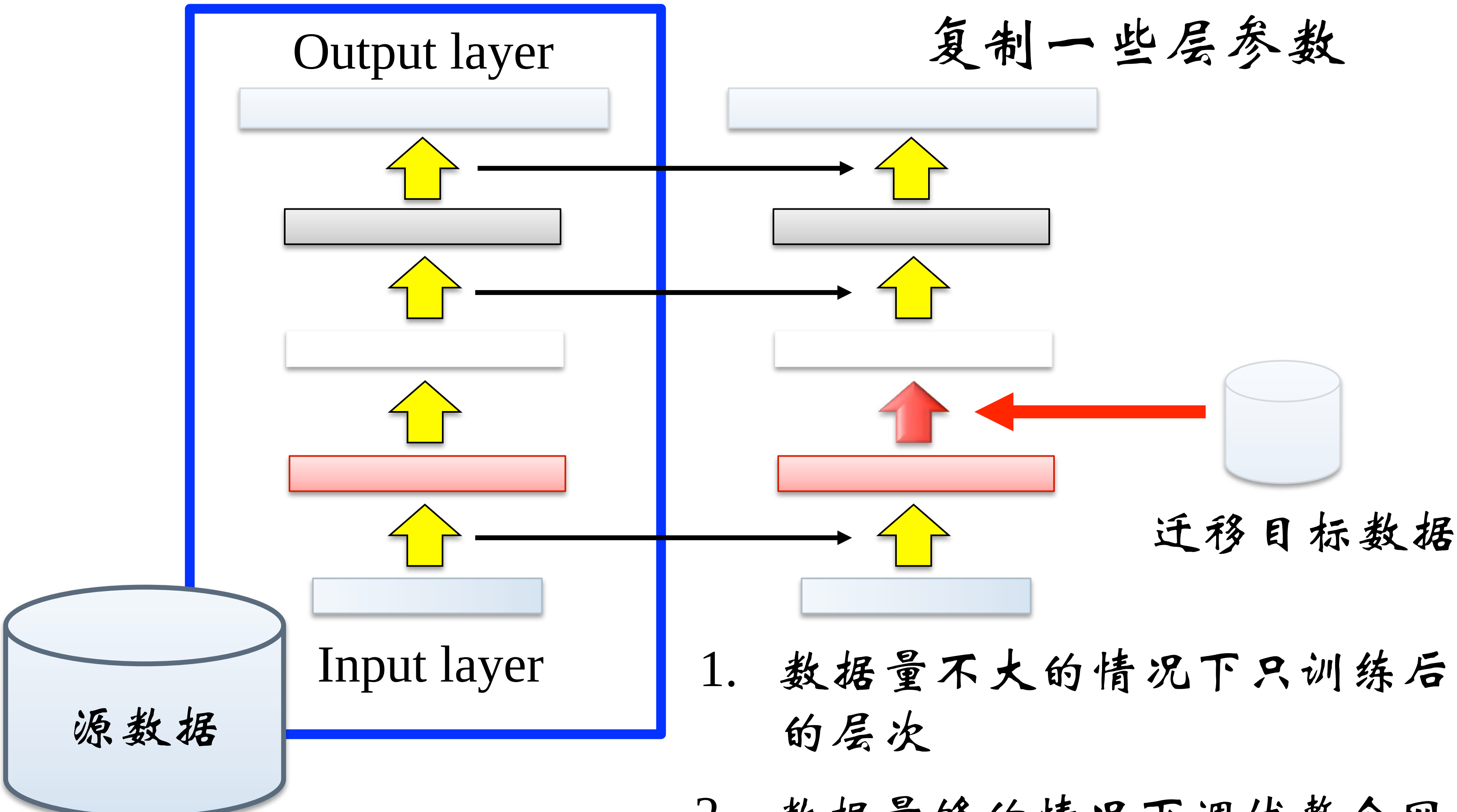
(Caffe + Matlab) https://github.com/daijifeng001/R-FCN

(Caffe + Python) https://github.com/Orpine/py-R-FCN

有监督到有监督：fine-tune



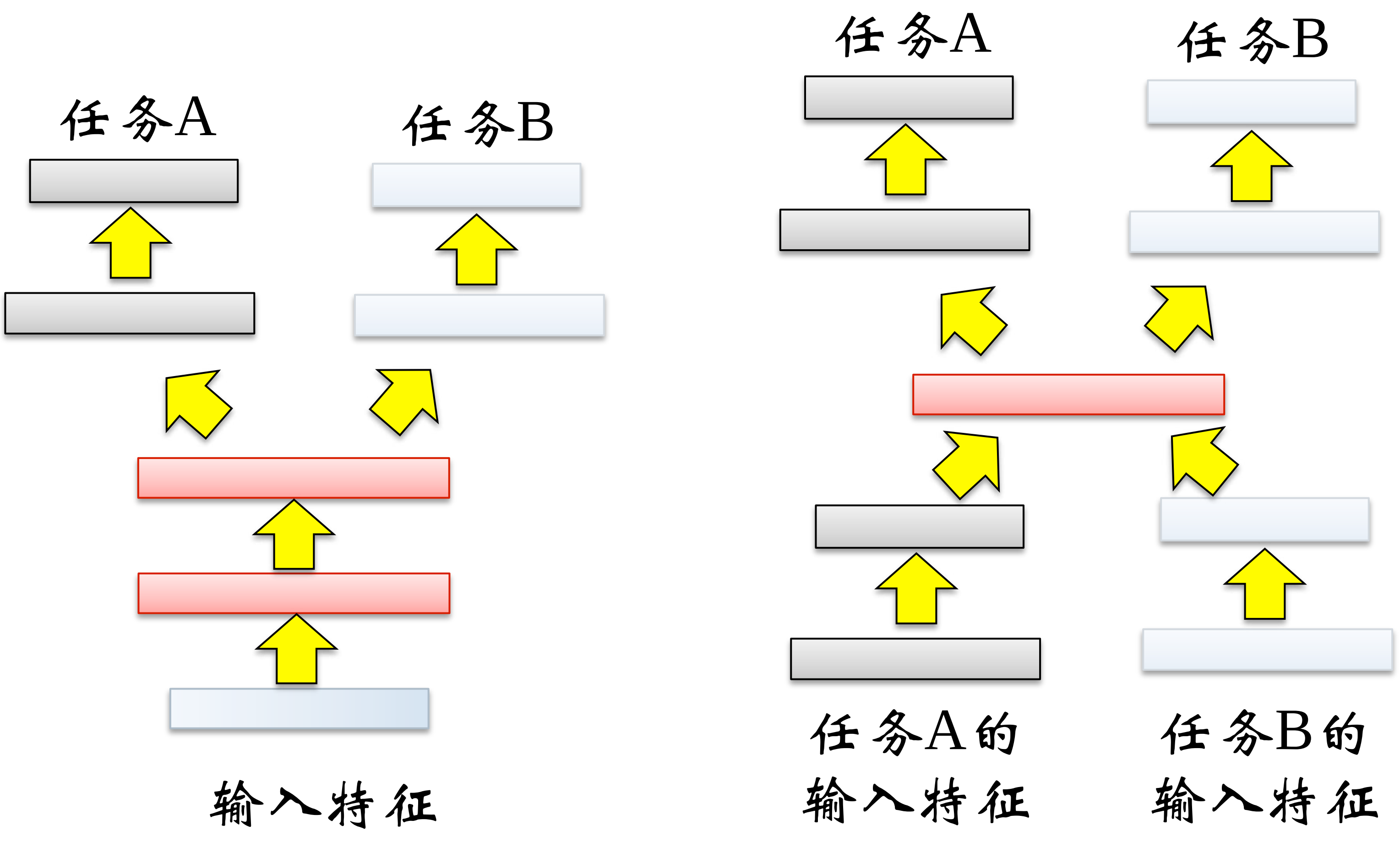
Layer Transfer



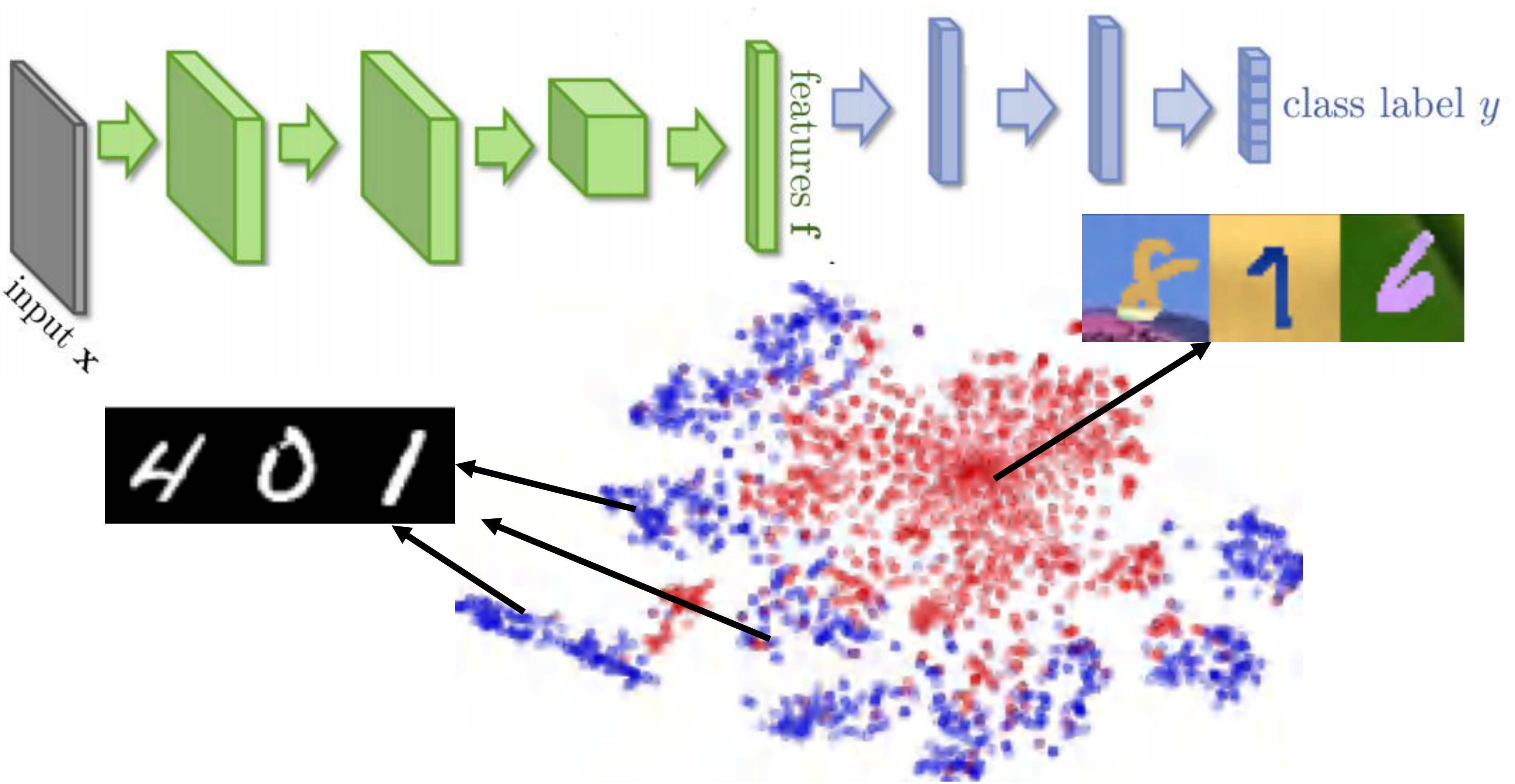
1. 数据量不大的情况下只训练后续的层次
2. 数据量够的情况下调优整个网络

Multitask Learning

- 神经网络是多层次的结构，适合多任务学习



有监督到无监督：域对抗学习

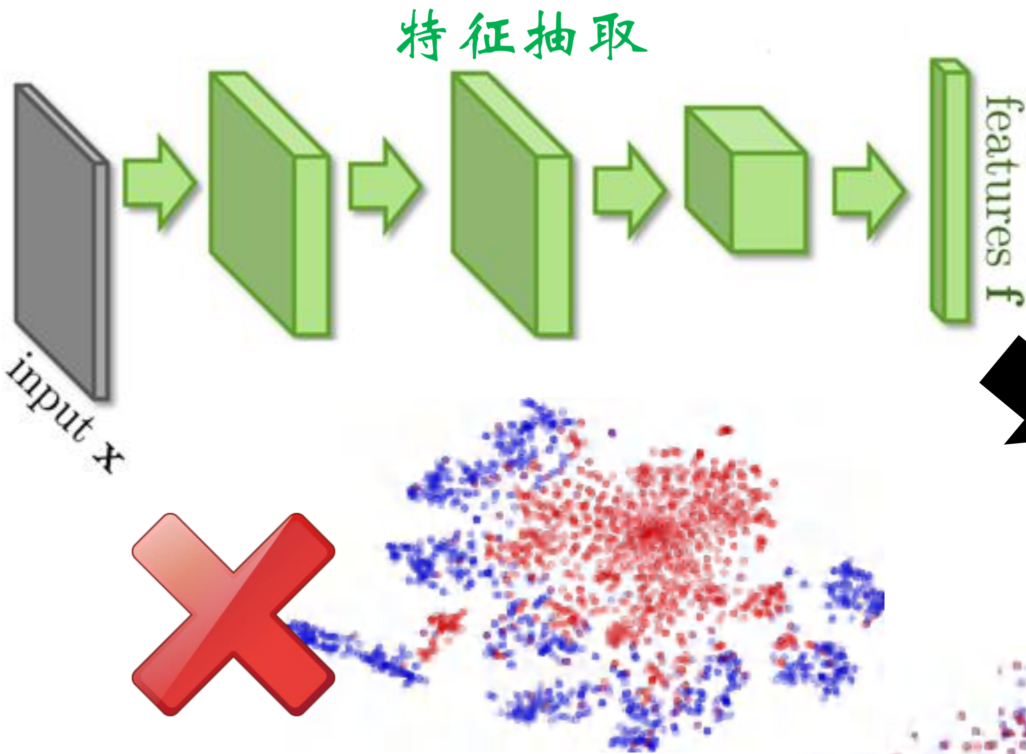


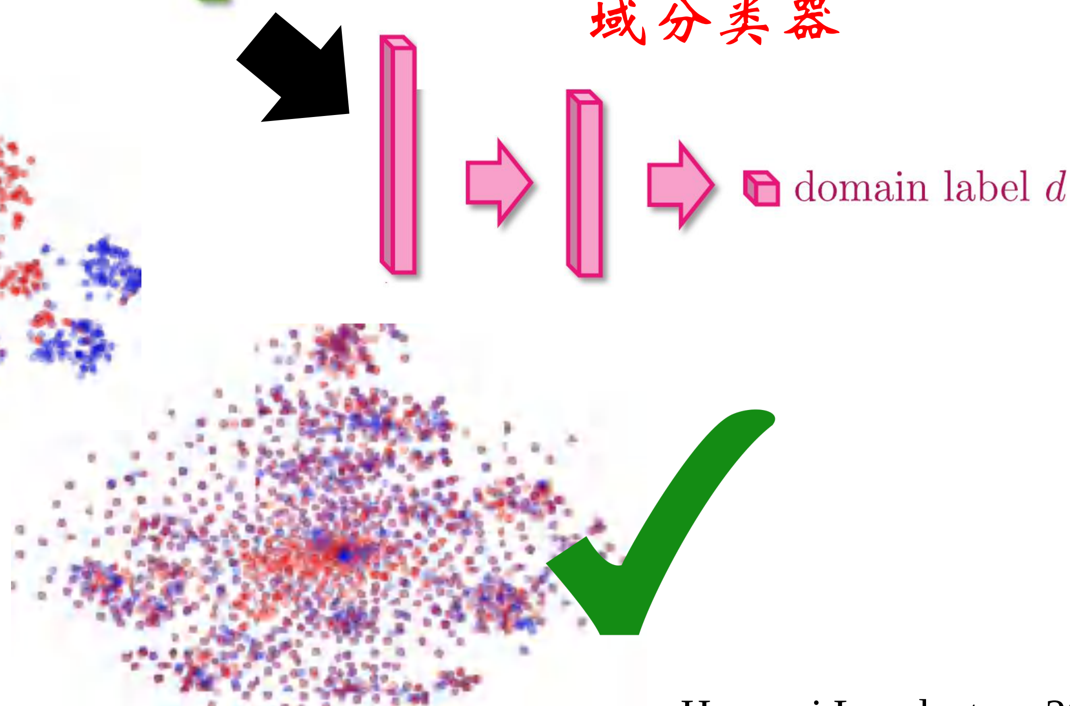
Hung-yi Lee lecture 2017

有监督到无监督：域对抗学习

类似于GAN

Too easy to feature extractor ……



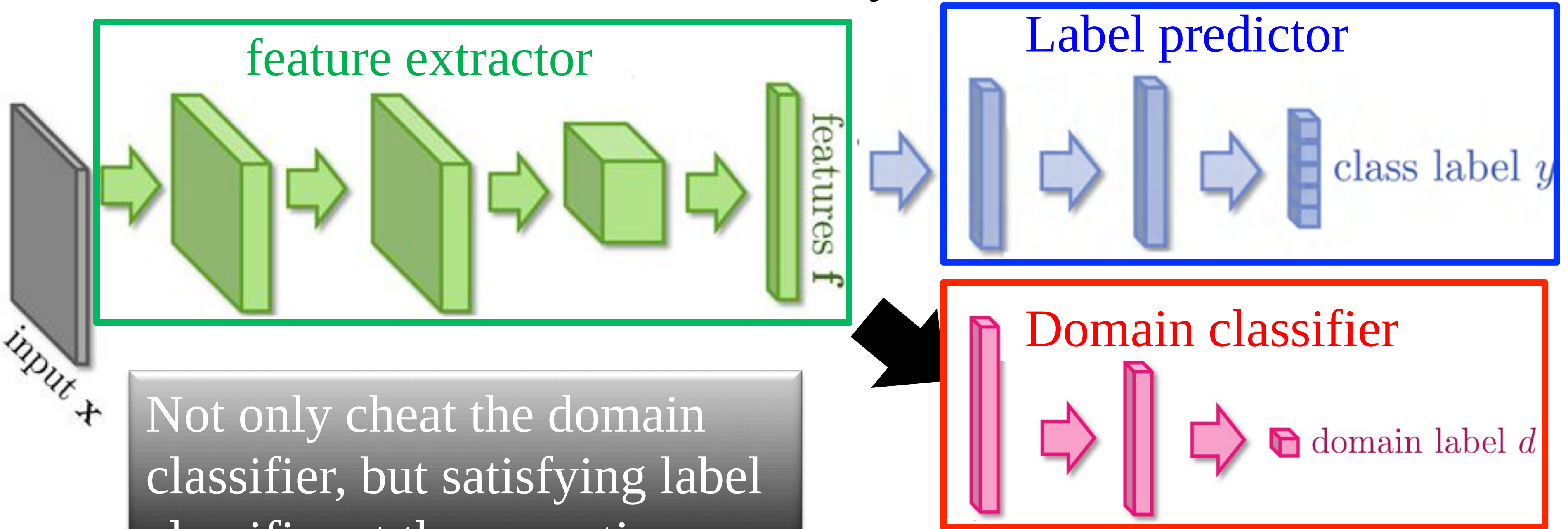


Hung-yi Lee lecture 2017

有监督到无监督：域对抗学习

Maximize label classification accuracy + minimize domain classification accuracy

Maximize label classification accuracy

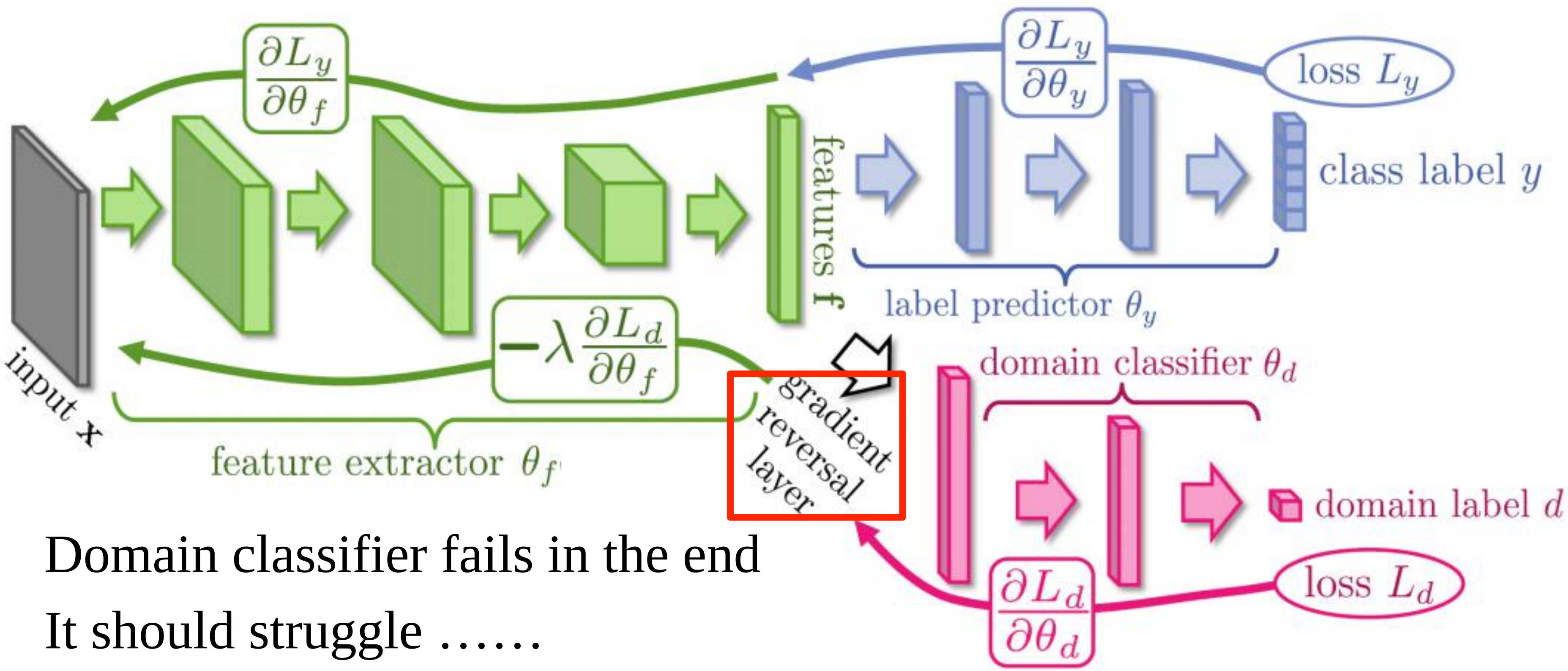


Not only cheat the domain classifier, but satisfying label classifier at the same time

Maximize domain classification accuracy

This is a big network, but different parts have different goals.

有监督到无监督：域对抗学习



Domain classifier fails in the end

It should struggle ……

Yaroslav Ganin, Victor Lempitsky, Unsupervised Domain Adaptation by Backpropagation, ICML, 2015

Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Domain-Adversarial Training of Neural Networks, JMLR, 2016

感谢大家！

恳请大家批评指正！

寒小阳
hanxiaoyang.ml@gmail.com