CNN理解、GAN、RNN、目标检测

主讲：邓伟洪
http://www.pris.net.cn/introduction/teacher/dengweihong
模式识别与智能系统实验室

人工智能学院
北京邮电大学

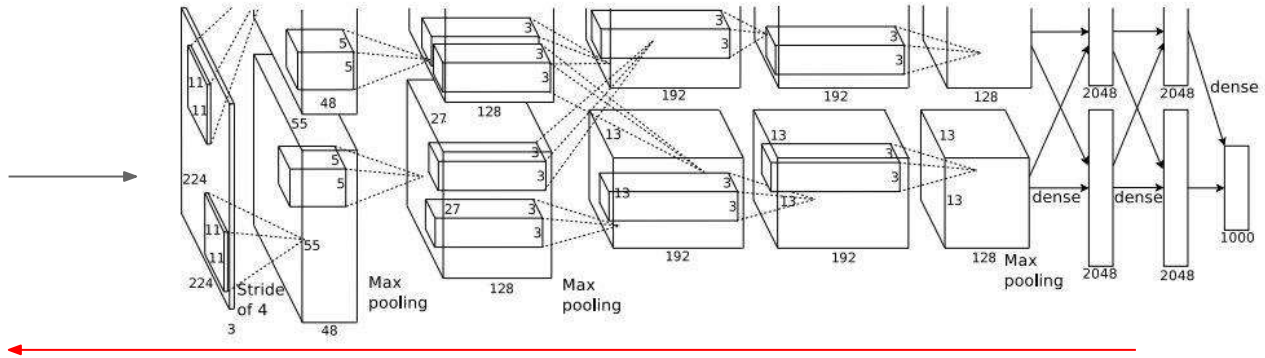
可视化卷积神经网络的特征：梯度上升

$$\arg\max_I S_c(I) - \lambda \|I\|_2^2$$

C类的分值 (Softmax之前)

1. 将图像初始化为0

零值图像



重复以下步骤：

2. 前向传播图像并计算当前分值
3. 通过反向传播计算相对于图像像素神经元分值的梯度
4. 对图像执行一个小的梯度上升更新

可视化卷积神经网络的特征：梯度上升

$$\arg\max_I S_c(I) - \lambda \|I\|_2^2$$

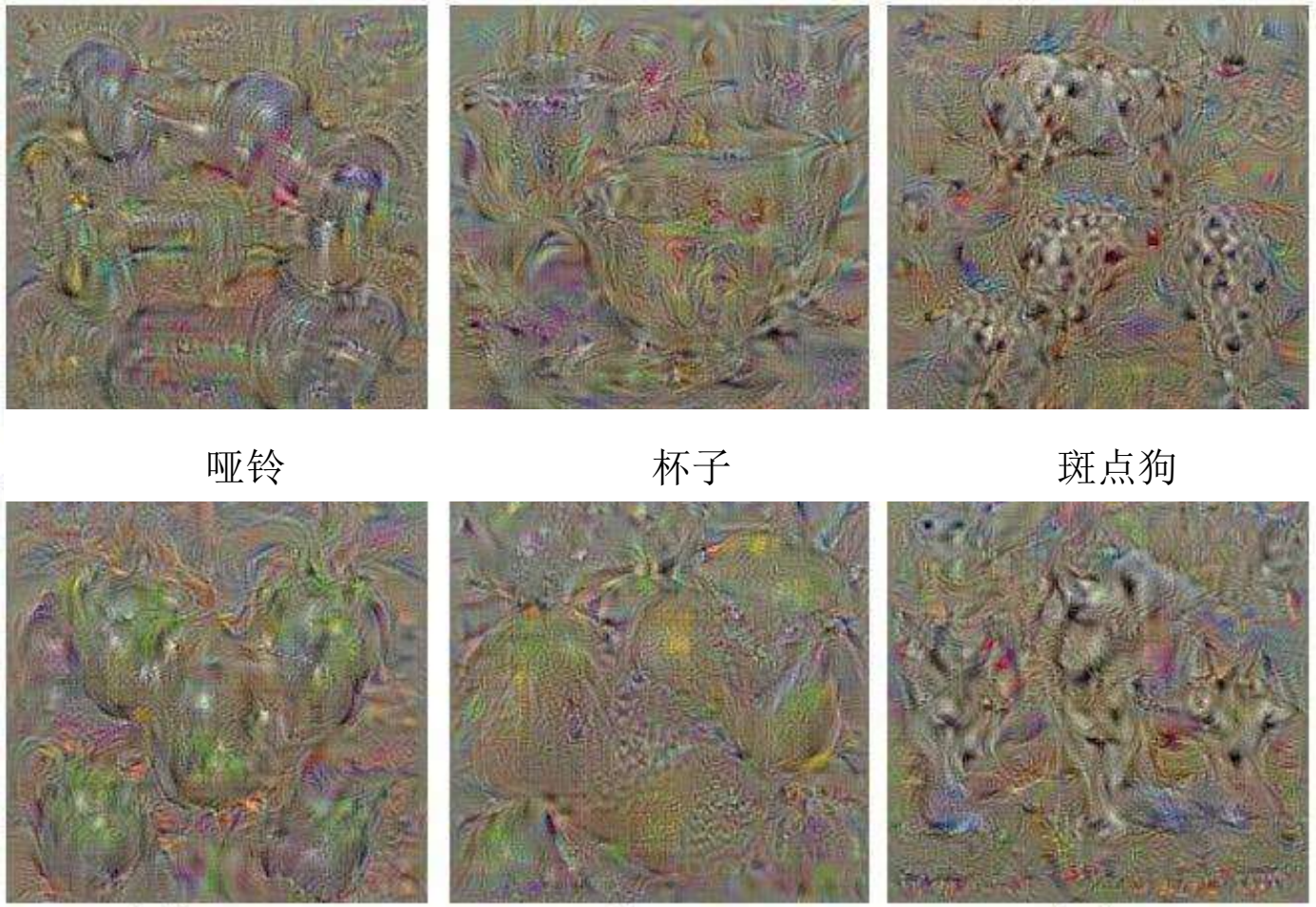
简单正则项：惩罚生成图像的L2范数

Simonyan, Vedaldi, and Zisserman, “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps”, ICLR Workshop 2014.

可视化卷积神经网络的特征：梯度上升

$$\arg\max_I S_c(I) - \lambda \|I\|_2^2$$

简单正则项：惩罚生成图像的L2范数

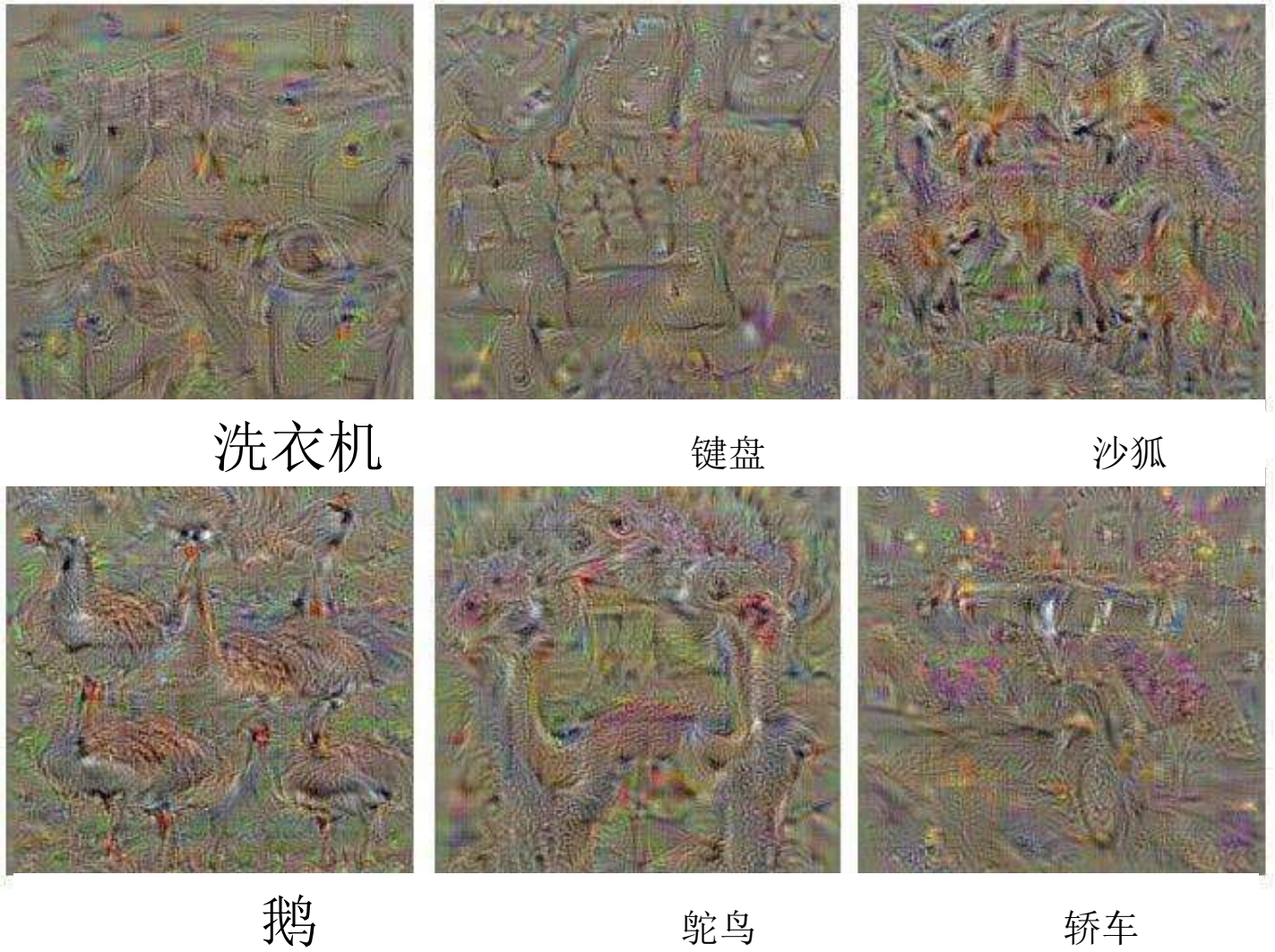


Simonyan, Vedaldi, and Zisserman, “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps”, ICLR Workshop 2014.

可视化卷积神经网络的特征：梯度上升

$$\arg\max_I S_c(I) - \lambda \|I\|_2^2$$

简单正则项：惩罚生成图像的L2范数

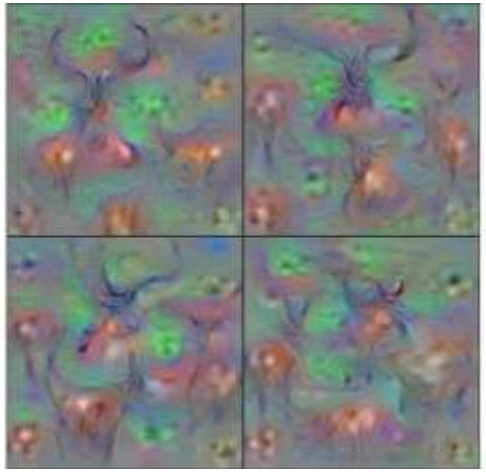


Yosinski et al, “Understanding Neural Networks Through Deep Visualization”, ICML DL Workshop 2014.
Figure copyright Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson, 2014. Reproduced with permission.

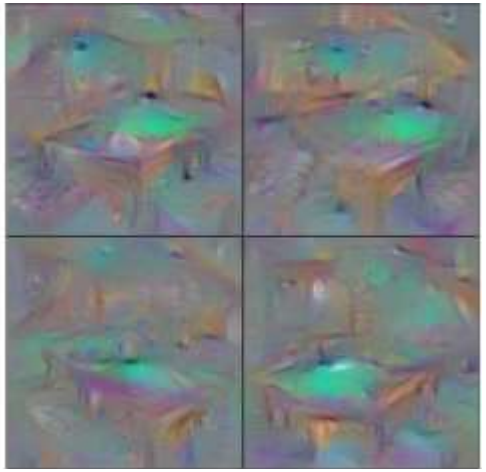
可视化卷积神经网络的特征：梯度上升

$$\arg\max_I S_c(I) - \lambda \|I\|_2^2$$

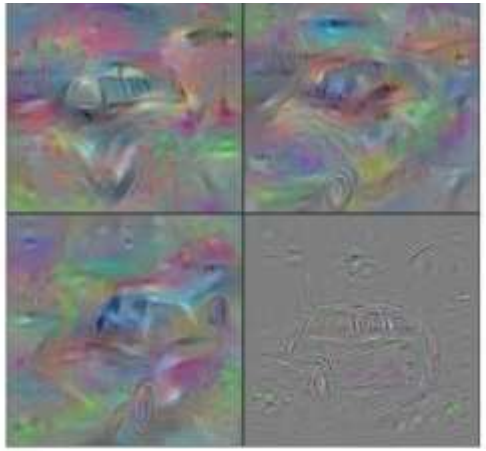
优化的正则项：惩罚图像的L2范数，
同时在优化过程中定期进行
(1) 高斯模糊处理
(2) 将小像素修改为0
(3) 将低梯度像素修改为0



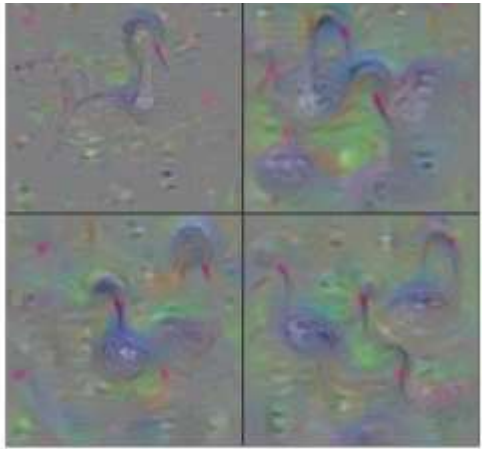
羚羊



台球桌



货车

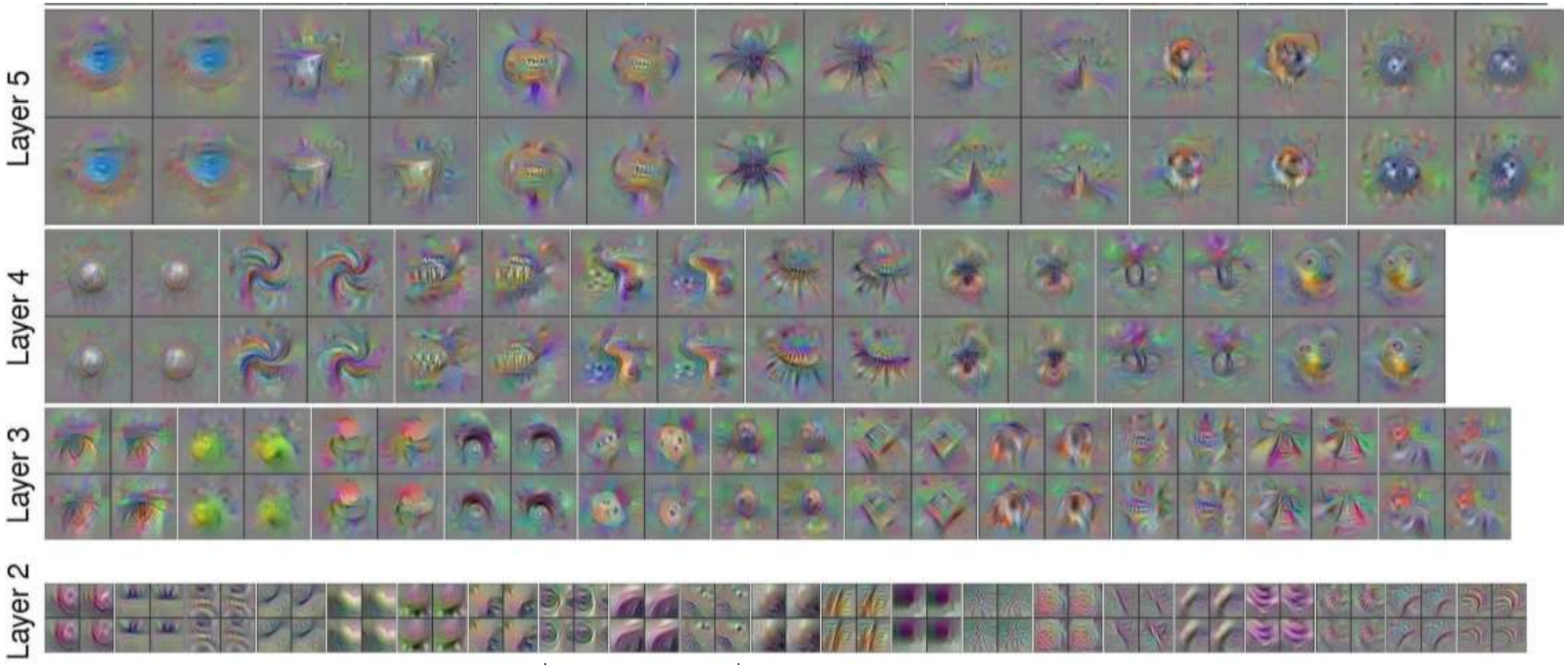


黑天鹅

Yosinski et al, “Understanding Neural Networks Through Deep Visualization”, ICML DL Workshop 2014.

可视化卷积神经网络的特征：梯度上升

用同样的方法可视化中间层特征



Yosinski et al, “Understanding Neural Networks Through Deep Visualization”, ICML DL Workshop 2014.

可视化卷积神经网络的特征：梯度上升

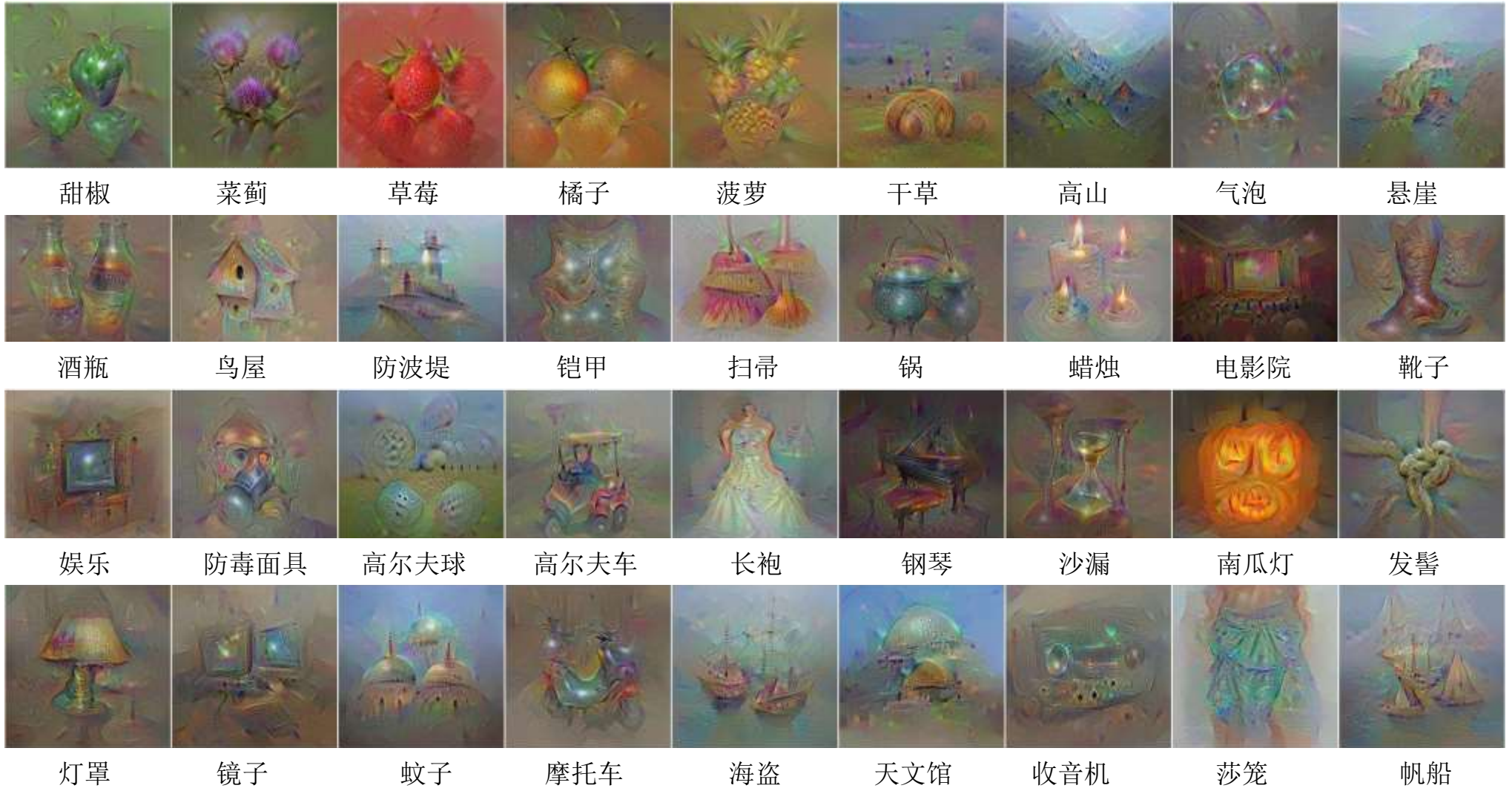
添加“多方位”可视化可以获得更好的结果
（加上优化的正则化，中心偏移）

重构“食品杂货店”神经元识别的多种特征类型

“食品杂货店”类中对应神经元识别的样本训练集图像

Nguyen et al, “Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks”, ICML Visualization for Deep Learning Workshop 2016.

可视化卷积神经网络的特征：梯度上升



Nguyen et al, “Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks”, ICML Visualization for Deep Learning Workshop 2016.

愚弄图像/对抗样本

(1) 任选一张图像
(2) 任选一个类别
(3) 最大化图像中该类别的分量
(4) 重复此过程直至网络错误分类

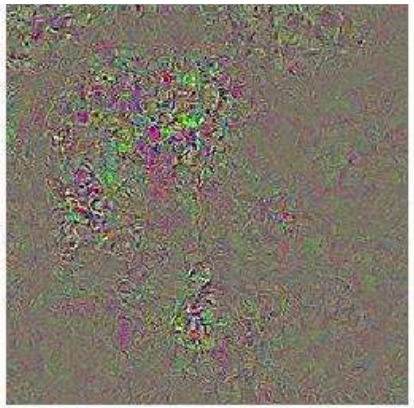
愚弄图像/对抗样本

非洲象

考拉

差异

10×差异

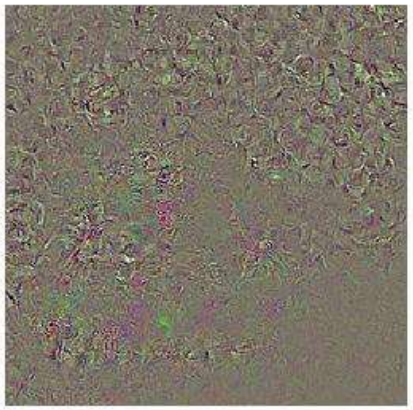


帆船

iPod

差异

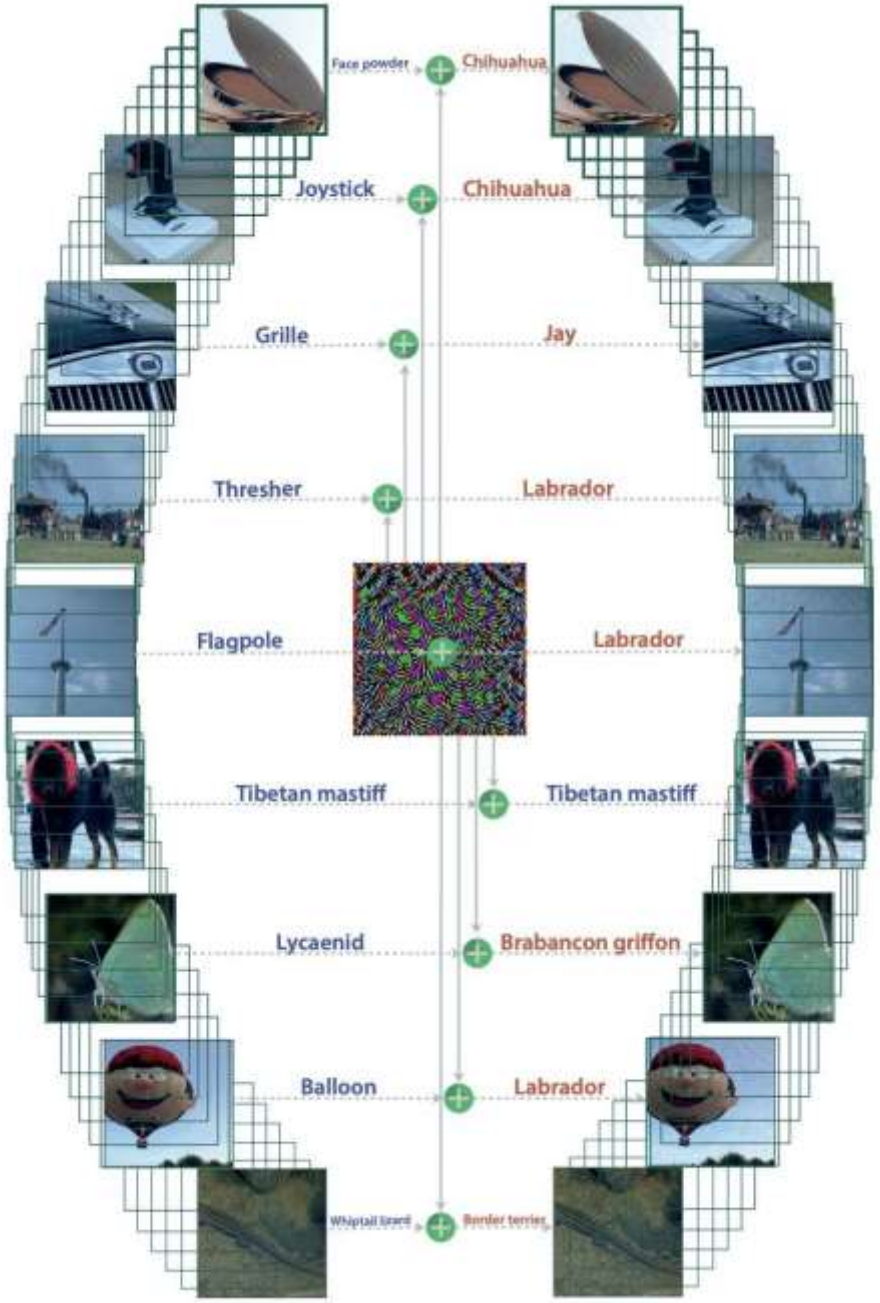
10×差异



Boat image is CC0 public domain
Elephant image is CC0 public domain

愚弄图像/对抗样本

通用扰动



Moosavi-Dezfooli, Seyed-Mohsen, et al. "Universal adversarial perturbations." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
Figure reproduced with permission

GAN的基本原理

GAN的学习方法

GAN的衍生模型

GAN的基本原理

GAN 的核心思想来源于博弈论的纳什均衡。它设定参与游戏双方分别为一个生成器 (Generator) 和一个判别器(Discriminator)，生成器的目的是尽量去学习真实的数据分布，而判别器的目的是尽量正确判别输入数据是来自真实数据还是来自生成器；为了取得游戏胜利，这两个游戏参与者需要不断优化， 各自提高自己的生成能力和判别能力，这个学习优化过程就是寻找二者之间的一个纳什均衡。

GAN的基本原理

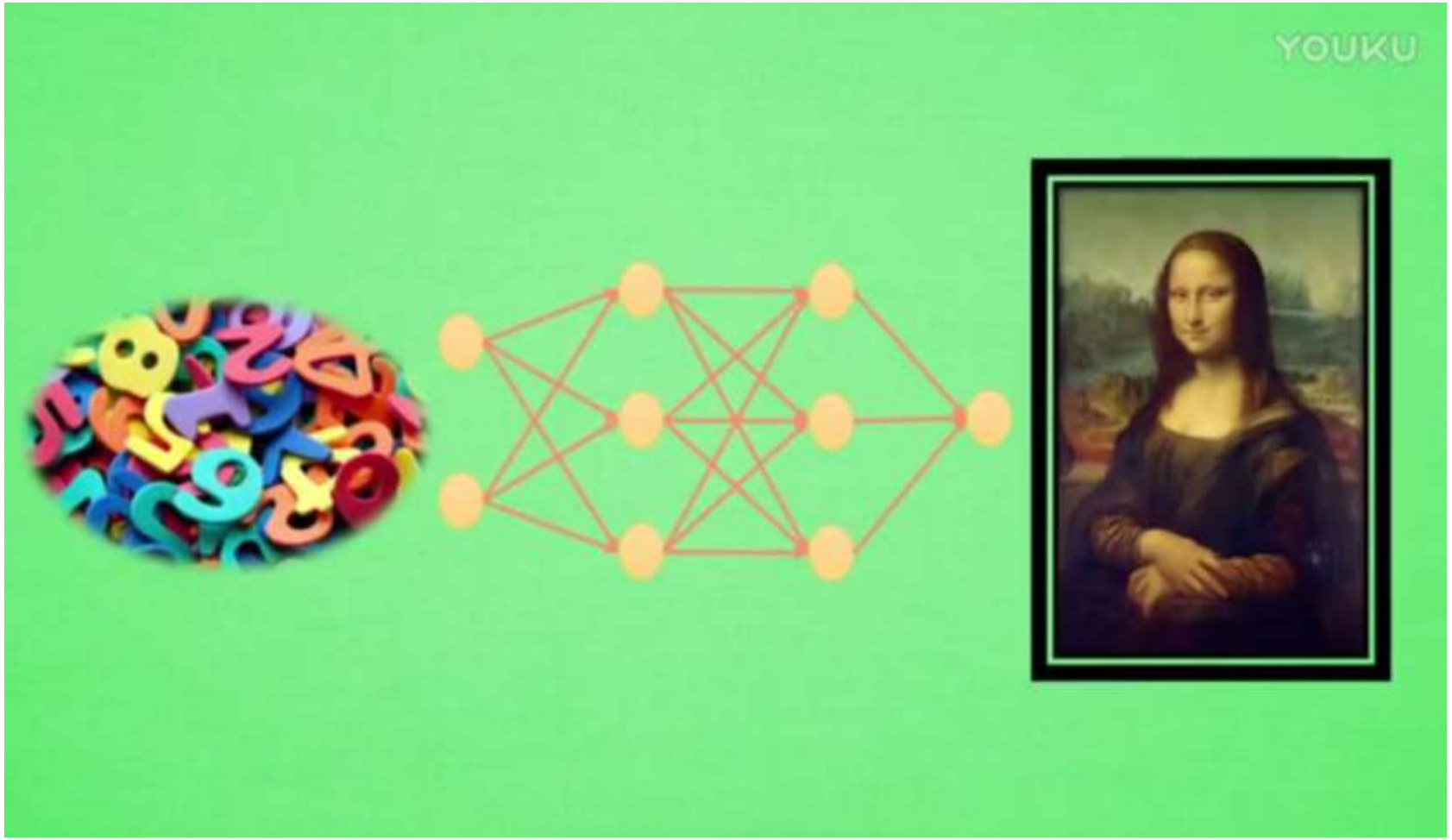


图3　生成网络模拟

GAN的基本原理

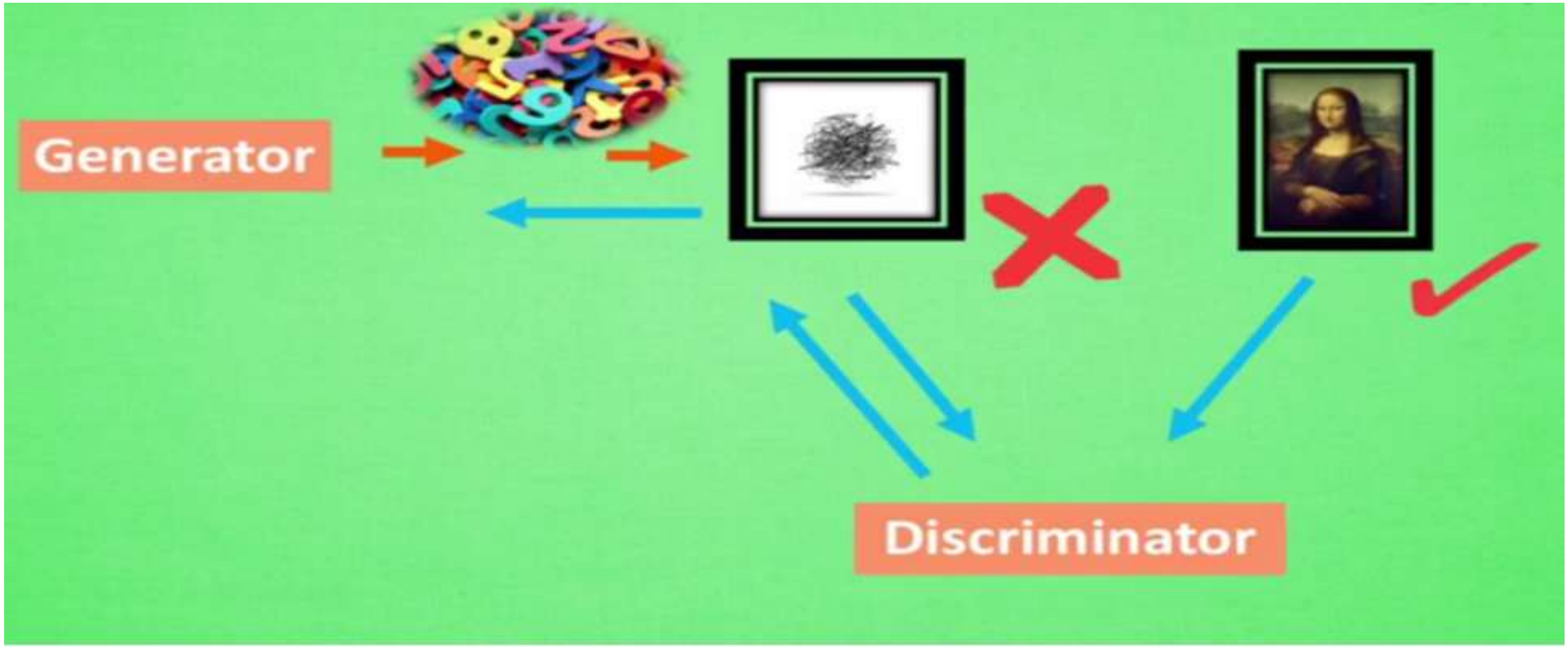
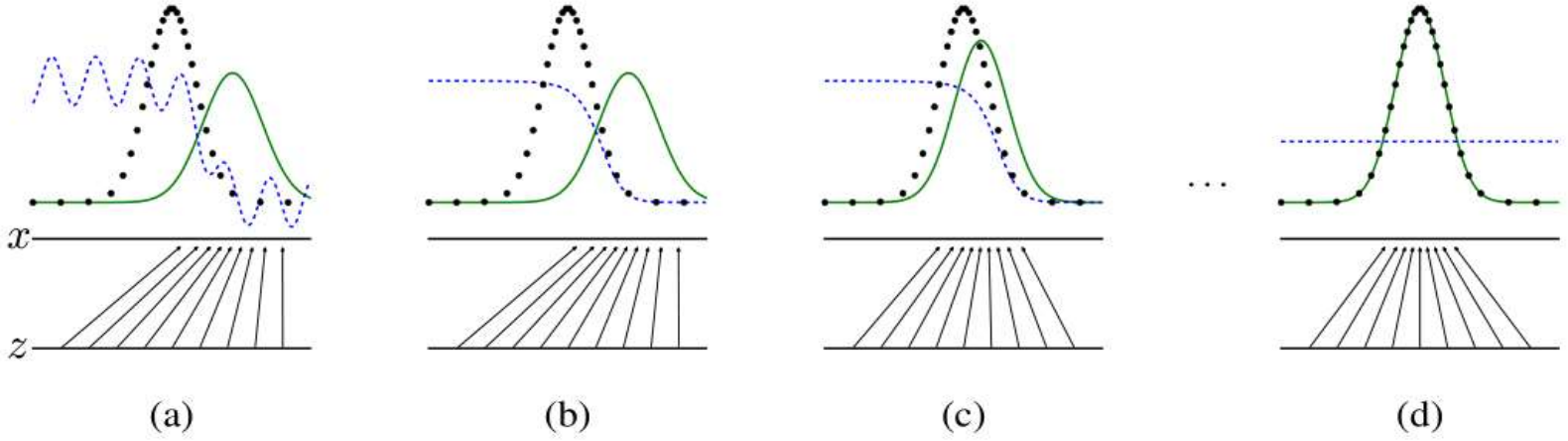


图4　GAN网络形象化流程

GAN的学习方法

首先，在给定生成器 G 的情况下，我们考虑最优化判别器 D.

$$\min_G \max_D \{f(D,G) = \mathrm{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathrm{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]\} \tag{5}$$



GAN的学习方法

总之，对于 GAN 的学习过程，我们需要训练模型 D 来最大化判别数据来源于真实数据或者伪数据分布 G(z) 的准确率，同时，我们需要训练模型 G来最小化 log(1 − D(G(z))).

这里可以采用交替优化的方法：先固定生成器 G，优化判别器 D，使得D 的判别准确率最大化；然后固定判别器 D，优化生成器 G，使得 D 的判别准确率最小化. 当 $p_{data} = p_g$ 时达到全局最优解. 训练 GAN 时，同一轮参数更新中，一般对 D 的参数更新 k 次再对 G的参数更新 1 次.

GAN的衍生模型

CGAN

DCGAN

Info GAN

……

WGAN

EBGAN

Improved GAN

GAN的衍生模型

（1）CGAN--条件生成对抗网络，为了防止训练崩塌将前置条件加入输入数据。

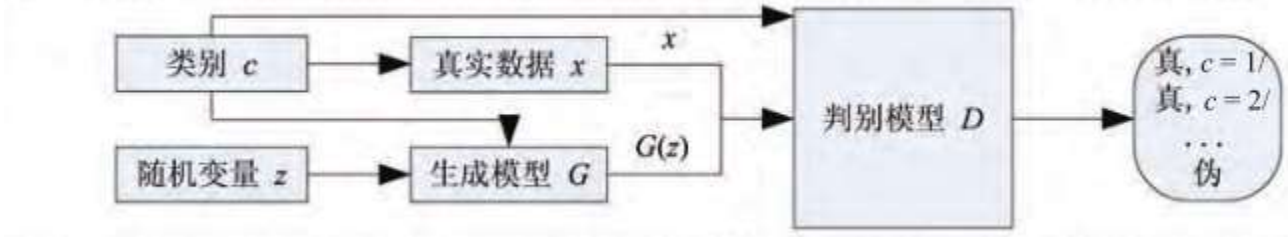


图5　条件生成对抗网络的结构

（2）DCGAN--深度卷积生成对抗网络，提出了能稳定训练的网络结构， 更易于工程实现。

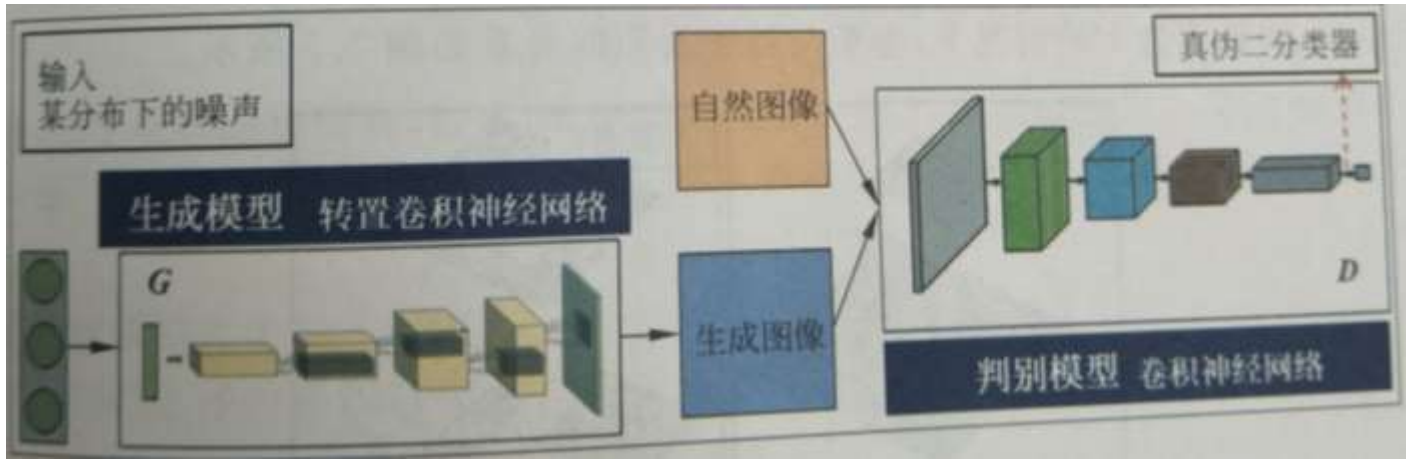


图6　深度卷积生成对抗网络的结构

GAN的衍生模型

（3）InfoGAN--信息最大化生成对抗网络，通过隐变量控制语义变化。

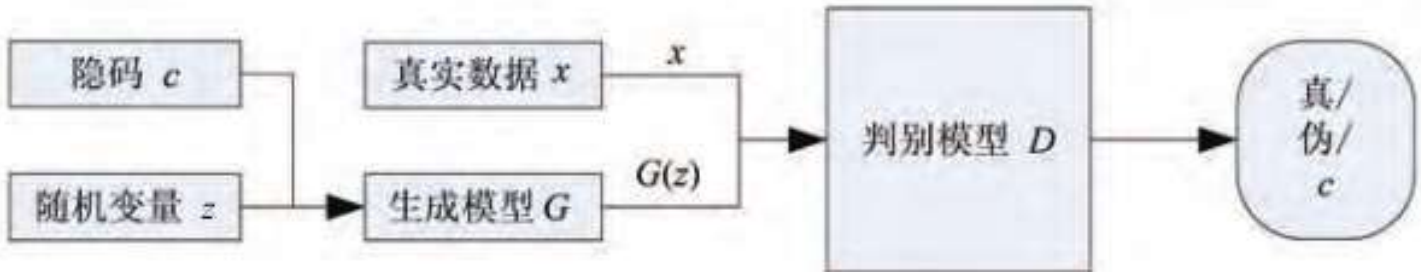


图7　InfoGAN的结构

（4）WGAN --定义了明确的损失函数，对 G&D 的距离给出了数学定义，较好地解决了训练坍塌问题。

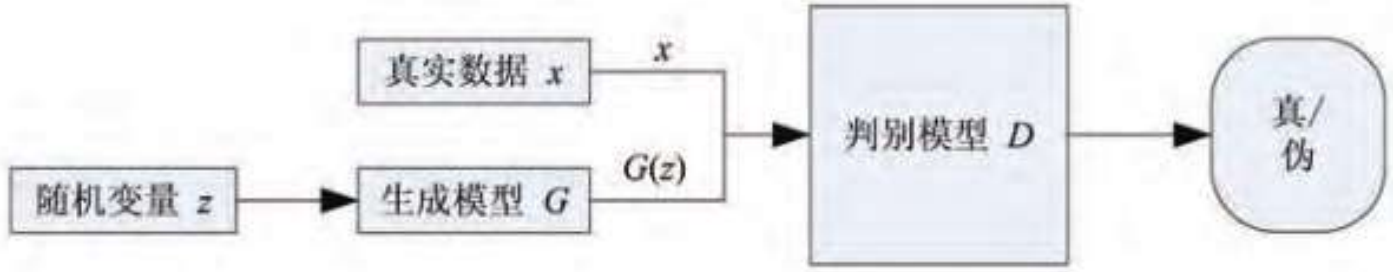


图8　WGAN的结构

GAN的衍生模型

（5） EBGAN--基于能量的生成式对抗网络，从能量模型角度给出了解释。

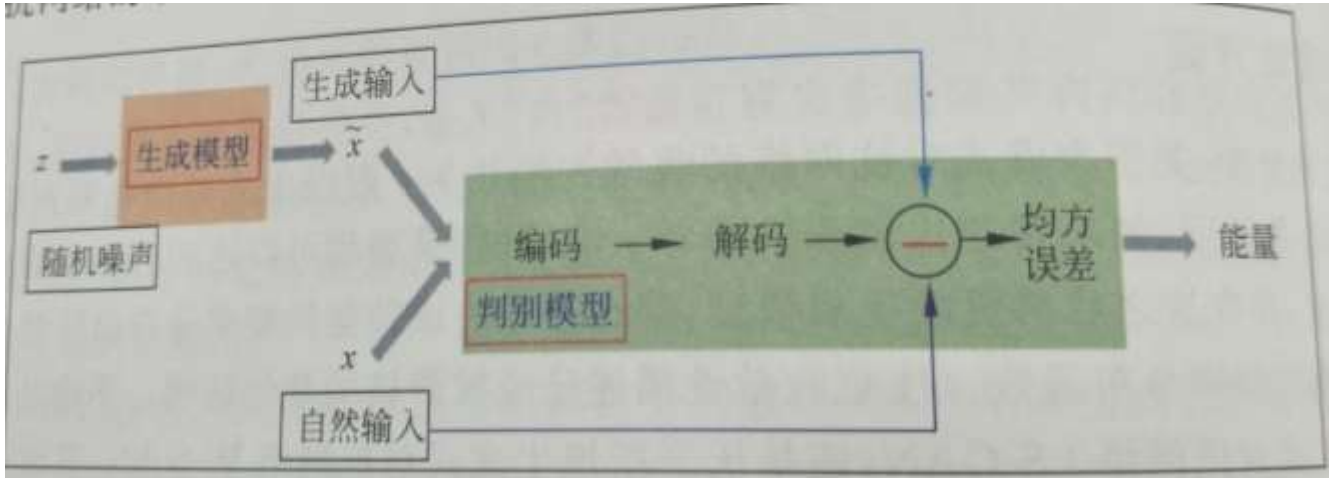


图9 EBGAN的结构

（6） Improved GAN--改进生成式对抗网络，提出了使模型训练稳定的五条经验。

a. 特征匹配（feature matching）
b. 最小批量判断（minibatch discrimination）
c. 历史平均（historical averaging）
d. 单边标签平滑（one-sided label smoothing）
e. 虚拟批量正则（virtual batch normalization）

GAN的应用

作为一个具有 “无限” 生成能力的模型, GAN的直接应用就是建模, 生成与真实数据分布一致的数据样本，GAN 可以用于解决标注数据不足时的学习问题。

其可以应用于：

- 图像和视觉领域
- 语音和语言领域
- 其他领域

图像和视觉领域

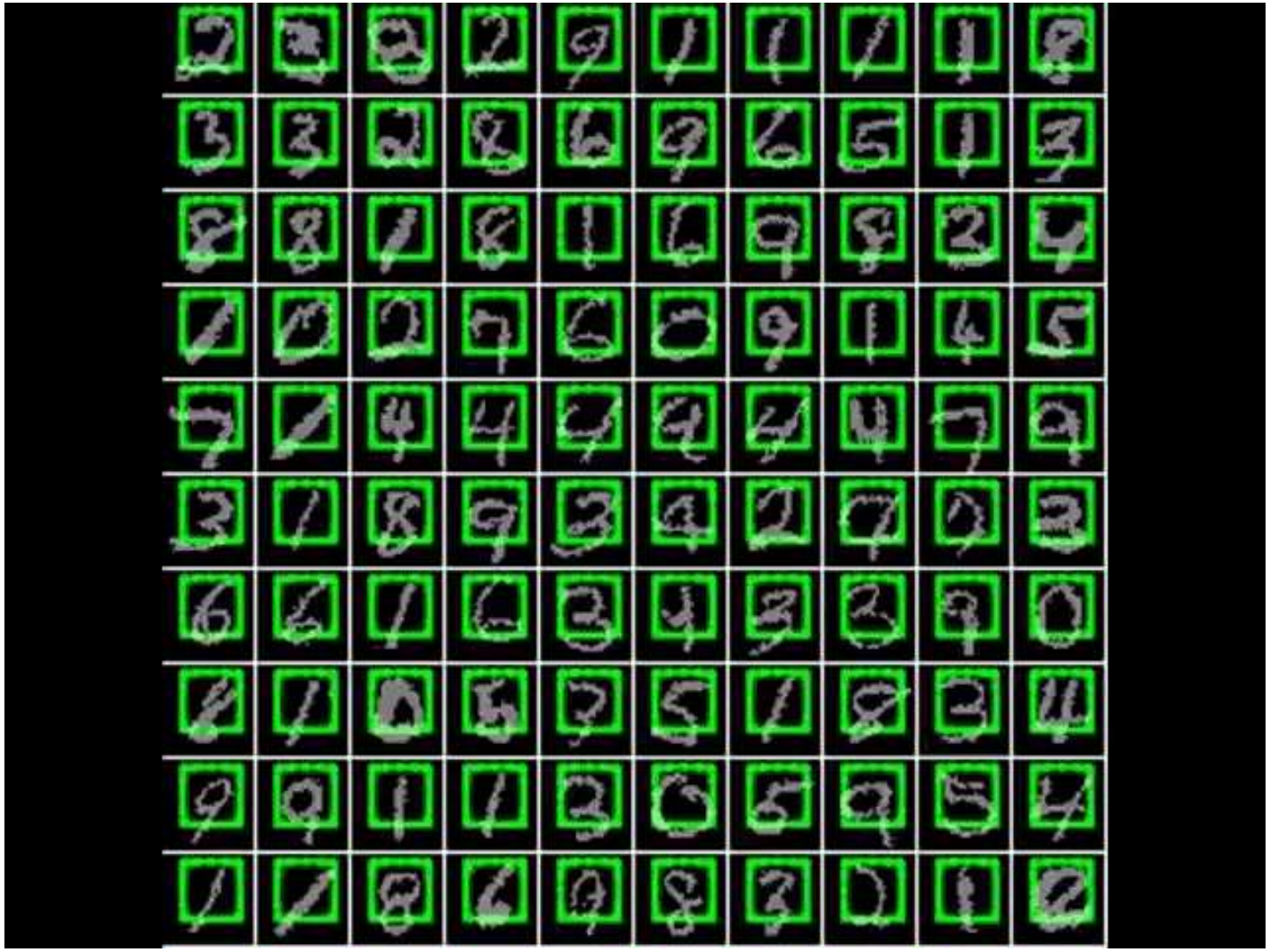
GAN 能够生成与真实数据分布一致的图像。一个典型应用是利用 GAN 来将一个低清模糊图像变换为具有丰富细节的高清图像。

用 VGG 网络作为判别器，用参数化的残差网络表示生成器，实验结果如图所示，可以看到 GAN 生成了细节丰富的图像。

图10　基于GAN的图像生成示例

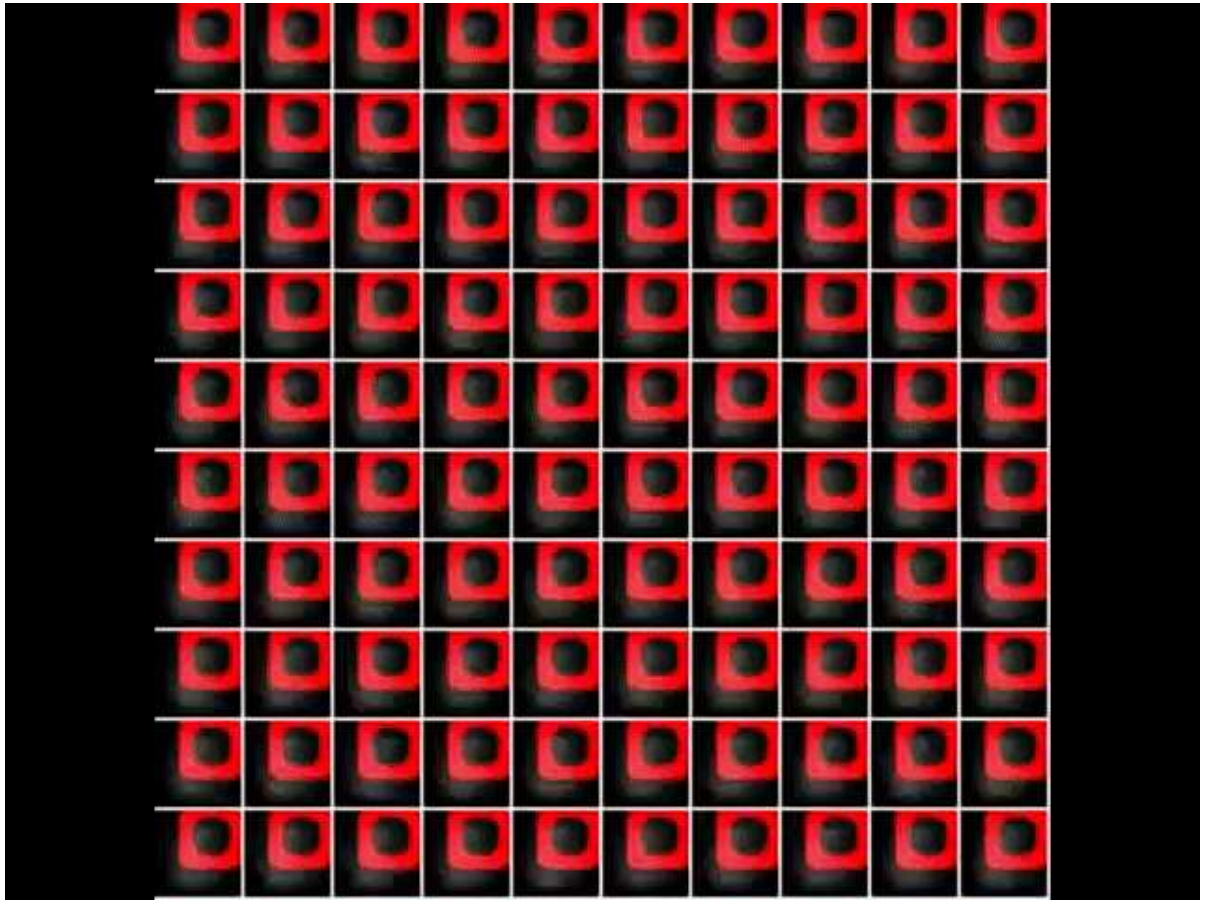
非序列数据的循序处理

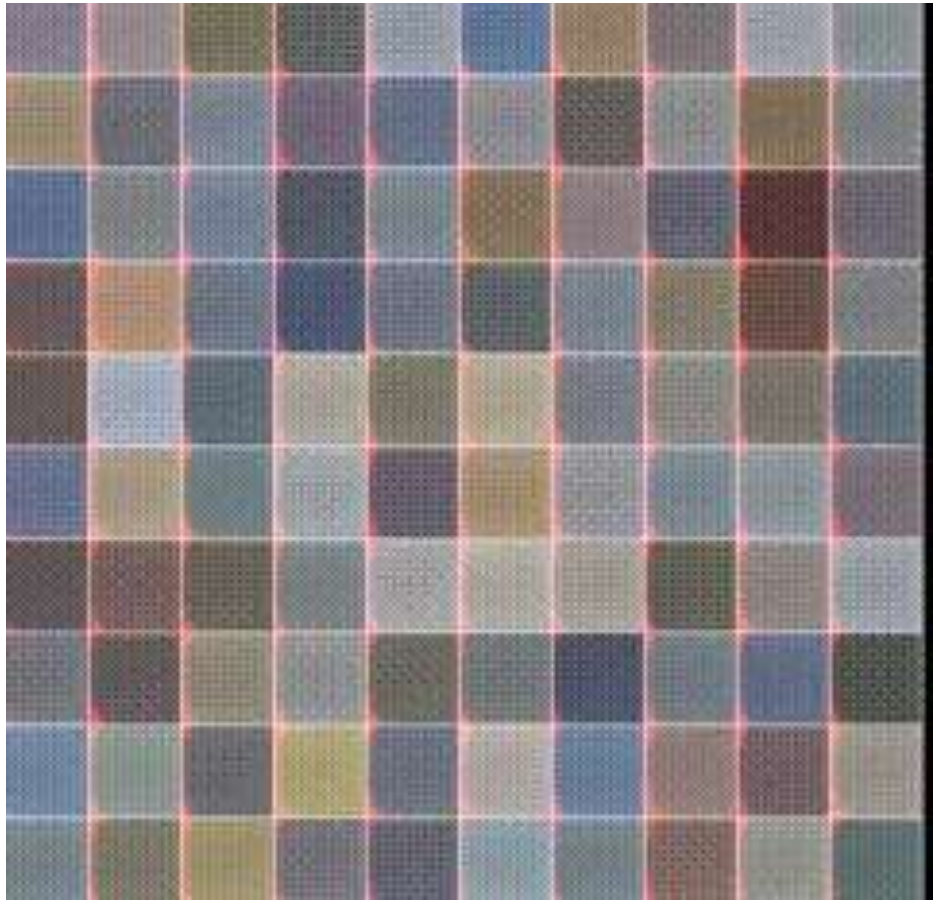
通过一系列的观察
来对图片分类



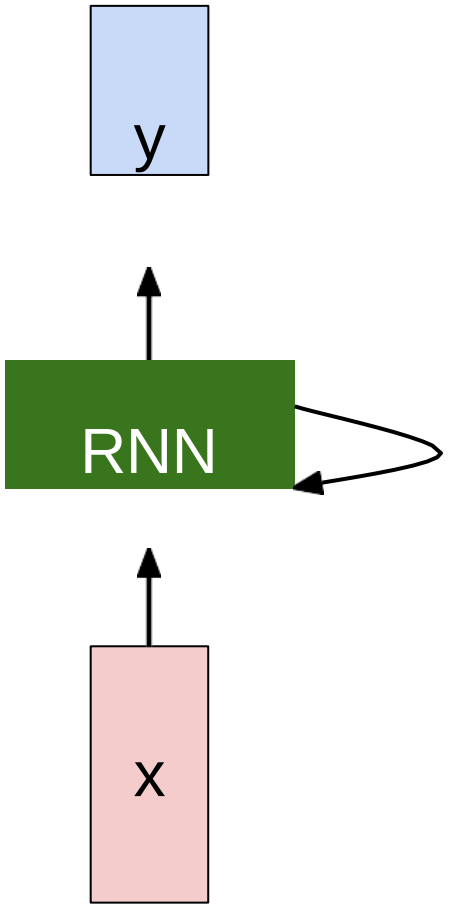
非序列数据的循序处理

一次生成一张图像!

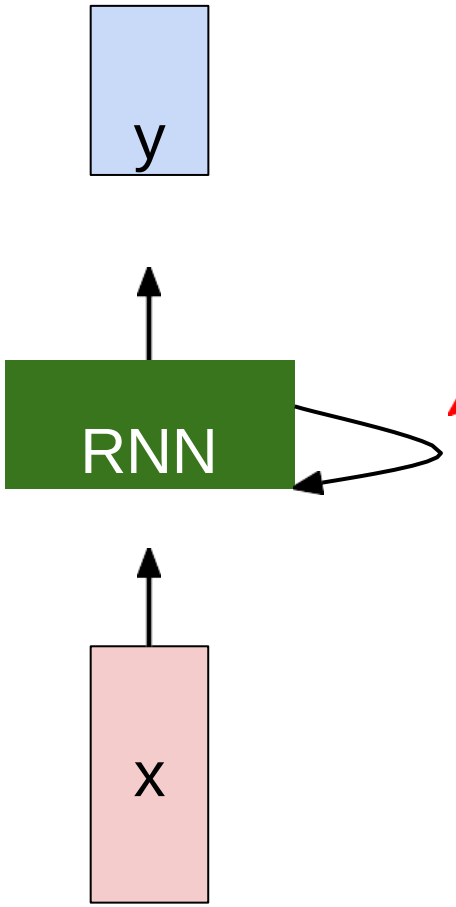




循环神经网络

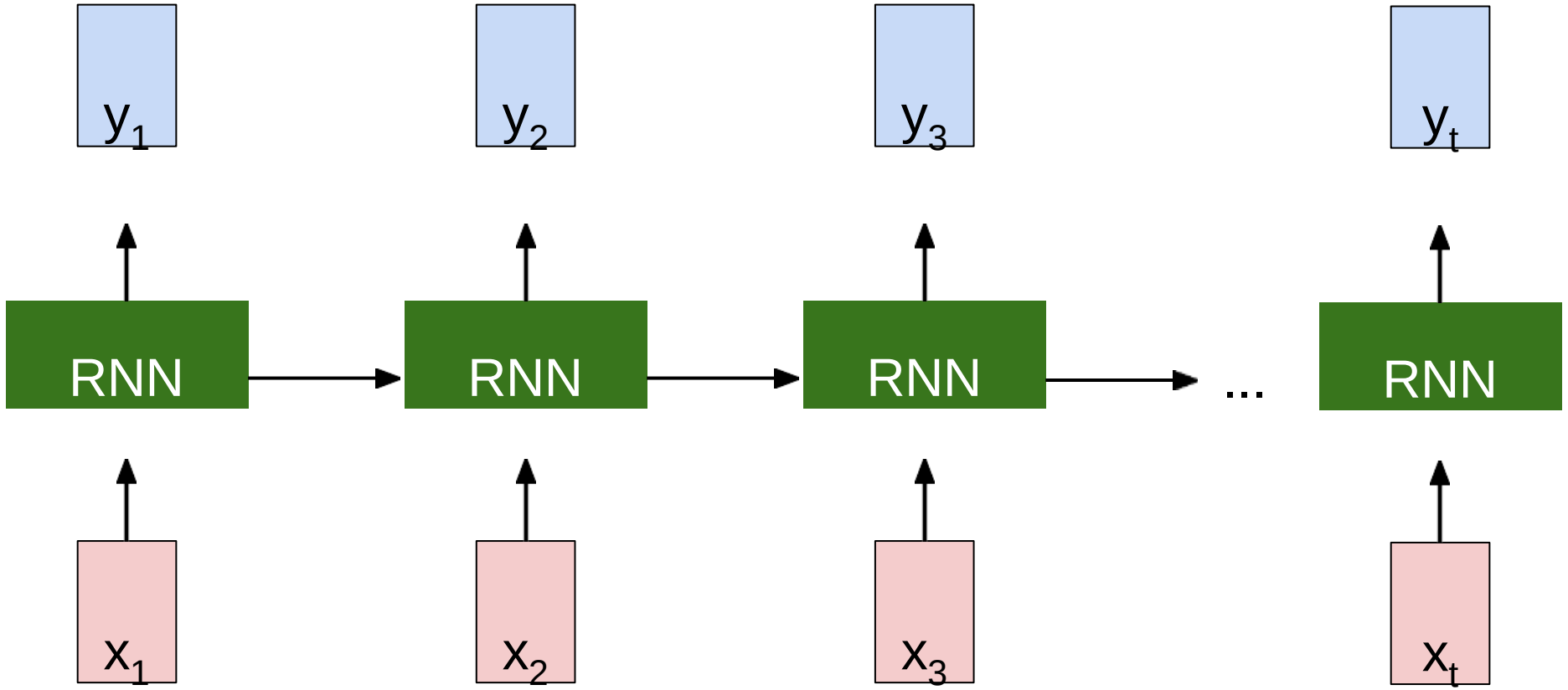


循环神经网络



关键思想：RNN有一个“内部状态”，随着网络对序列的处理而更新

循环神经网络



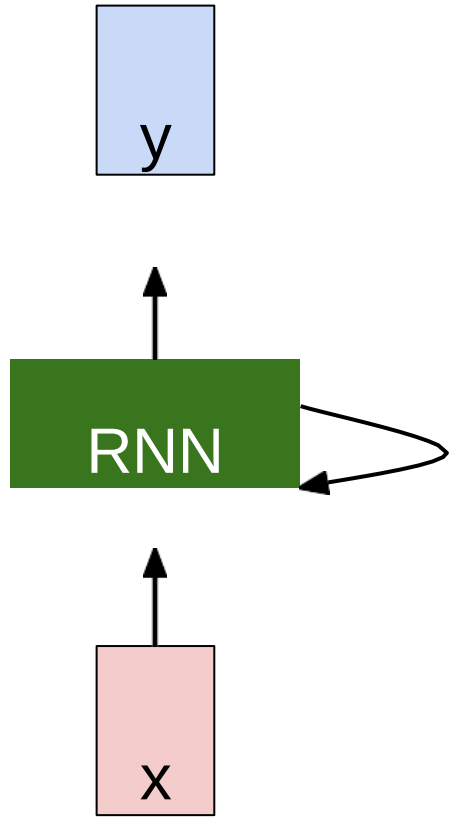
循环神经网络

我们可以通过在每个时间步应用**递归公式**来处理向量**x**序列：

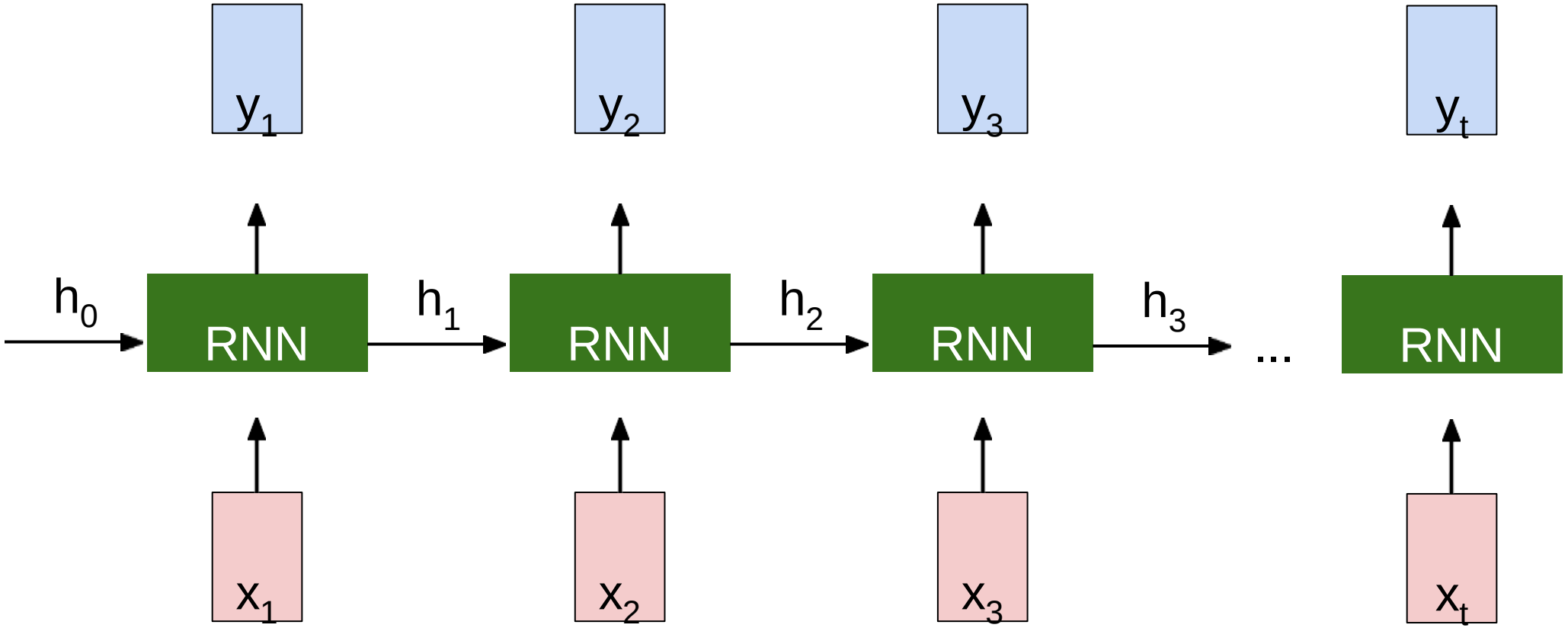
$$h_t = f_W(h_{t-1}, x_t)$$

新状态　　旧状态　　某一时间步的输入向量

参数为W的函数



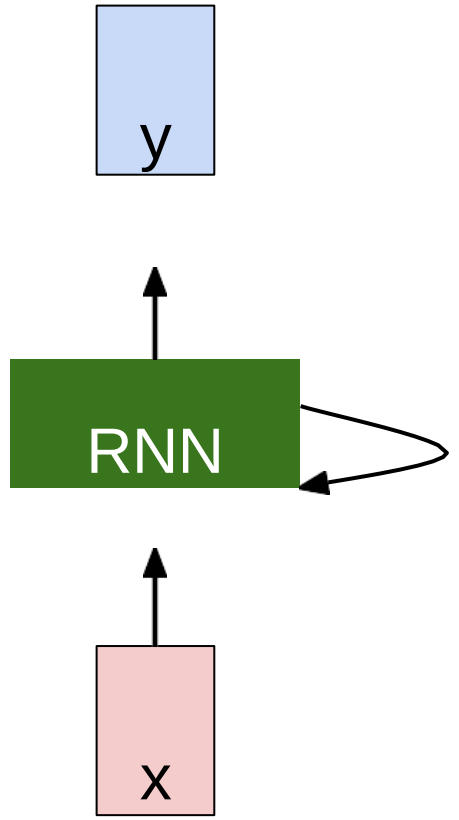
循环神经网络



循环神经网络

我们可以通过在每个时间步应用**递归公式**来处理向量**x**序列：

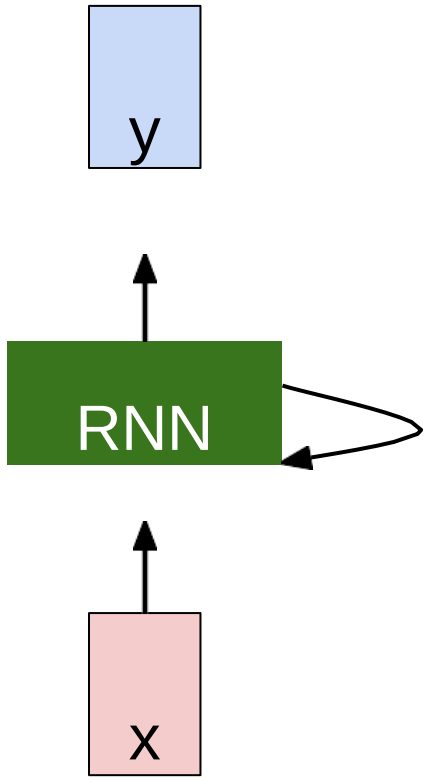
$$h_t = f_W(h_{t-1}, x_t)$$



注意：每个时间步使用一系列相同的函数和参数。

循环神经网络

状态仅由一个”隐藏”向量h构成:



$$h_t = f_W(h_{t-1}, x_t)$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

英文中有时称为“Vanilla RNN” 或者
“Elman RNN”

测试图像

This image is CC0 public domain

image

conv-64

conv-64

maxpool

conv-128

conv-128

maxpool

conv-256

conv-256

maxpool

conv-512

conv-512

maxpool

conv-512

conv-512

maxpool

FC-4096

FC-4096

FC-1000

softmax

测试图像

image

conv-64

conv-64

maxpool

conv-128

conv-128

maxpool

conv-256

conv-256

maxpool

conv-512

conv-512

maxpool

conv-512

conv-512

maxpool

FC-4096

FC-4096

FC-1000

softmax

X

测试图像

image

conv-64

conv-64

maxpool

conv-128

conv-128

maxpool

conv-256

conv-256

maxpool

conv-512

conv-512

maxpool

conv-512

conv-512

maxpool

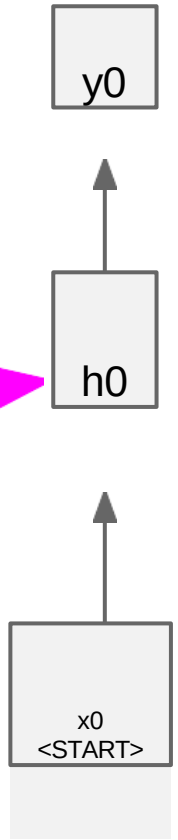
FC-4096

FC-4096

测试图像

x0
<START>

测试图像



Wih

v

之前:

$h = \tanh(W_{xh} * x + W_{hh} * h)$

现在:

$h = \tanh(W_{xh} * x + W_{hh} * h + \mathbf{W_{ih} * v})$

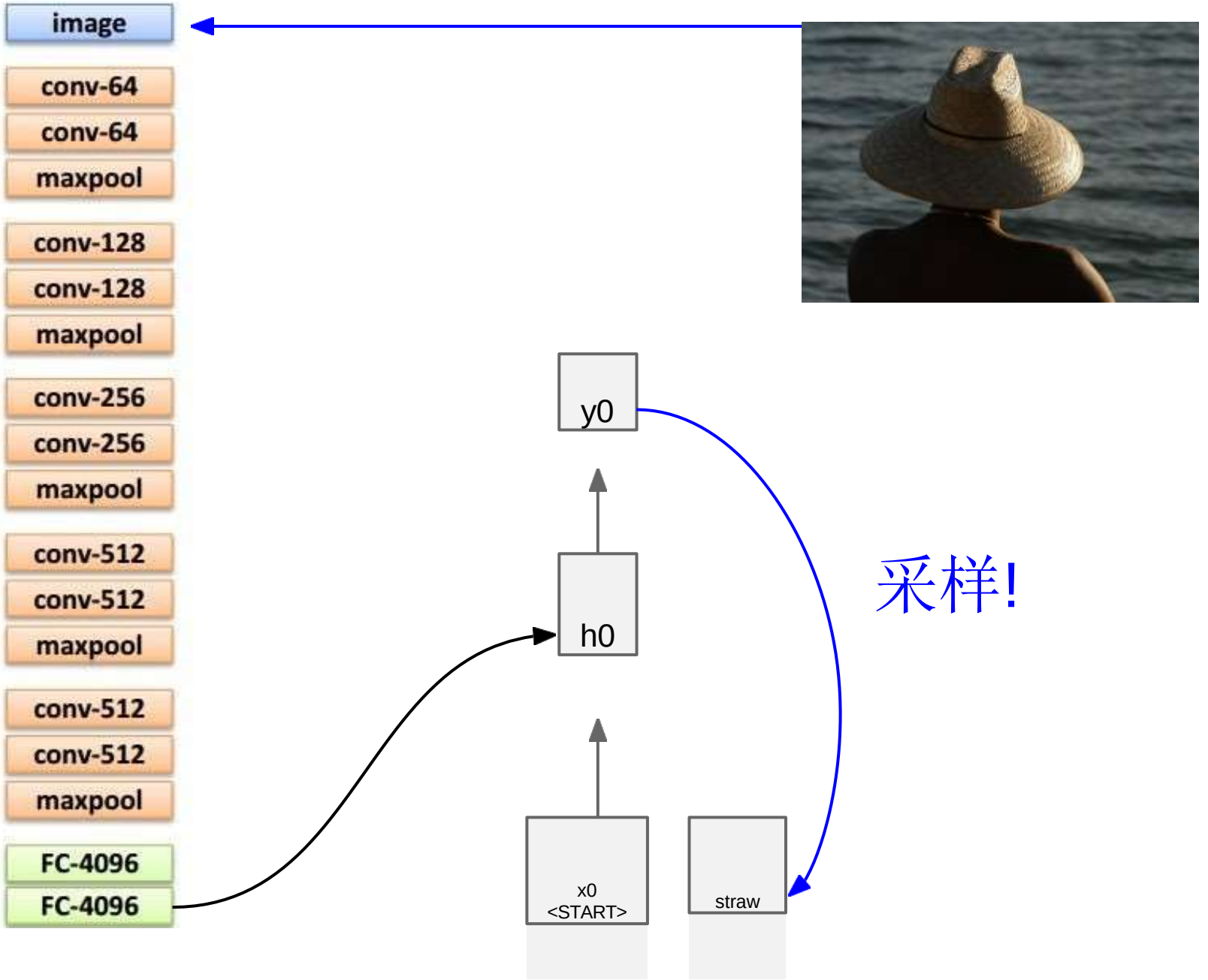


image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
maxpool
conv-512
conv-512
maxpool
conv-512
conv-512
maxpool
FC-4096
FC-4096
y0
h0
x0
<START>
straw
测试图像
采样!

image

conv-64
conv-64
maxpool

conv-128
conv-128
maxpool

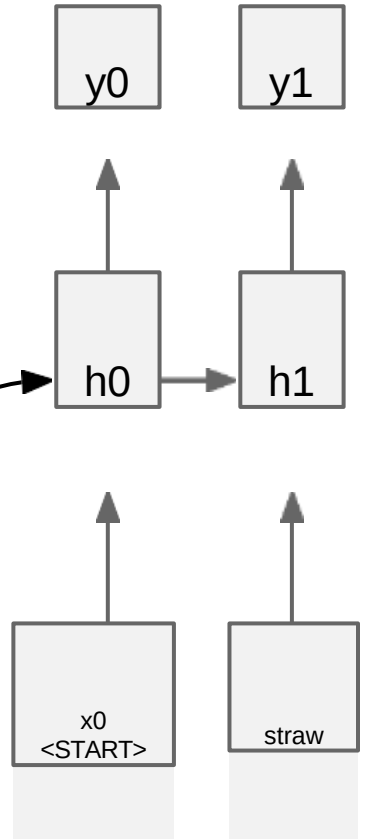
conv-256
conv-256
maxpool

conv-512
conv-512
maxpool

conv-512
conv-512
maxpool

FC-4096
FC-4096

测试图像



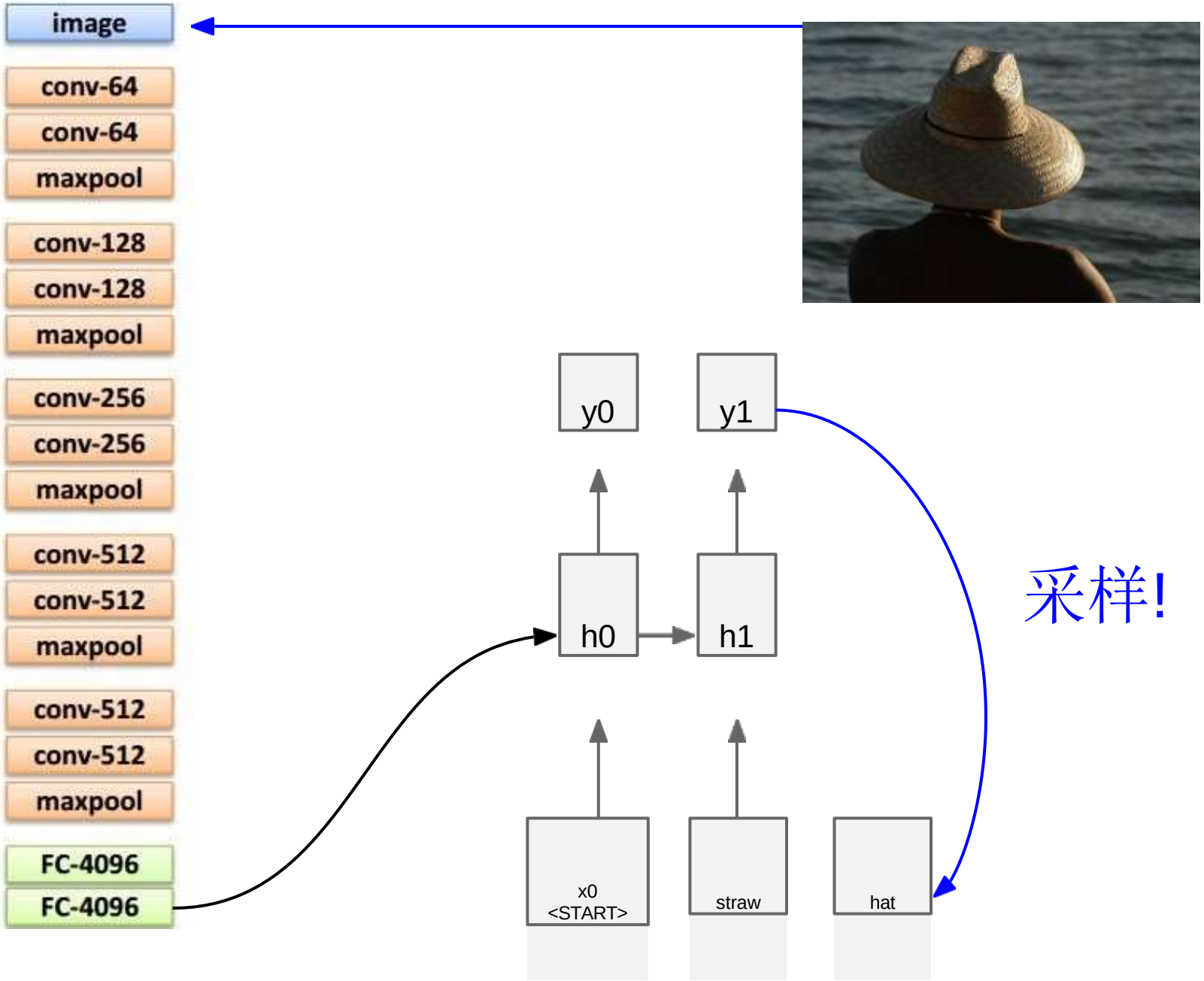


image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
maxpool
conv-512
conv-512
maxpool
conv-512
conv-512
maxpool
FC-4096
FC-4096
y0
y1
h0
h1
x0
<START>
straw
hat

测试图像

采样!

image

conv-64

conv-64

maxpool

conv-128

conv-128

maxpool

conv-256

conv-256

maxpool

conv-512

conv-512

maxpool

conv-512

conv-512

maxpool

FC-4096

FC-4096

测试图像

y0 y1 y2

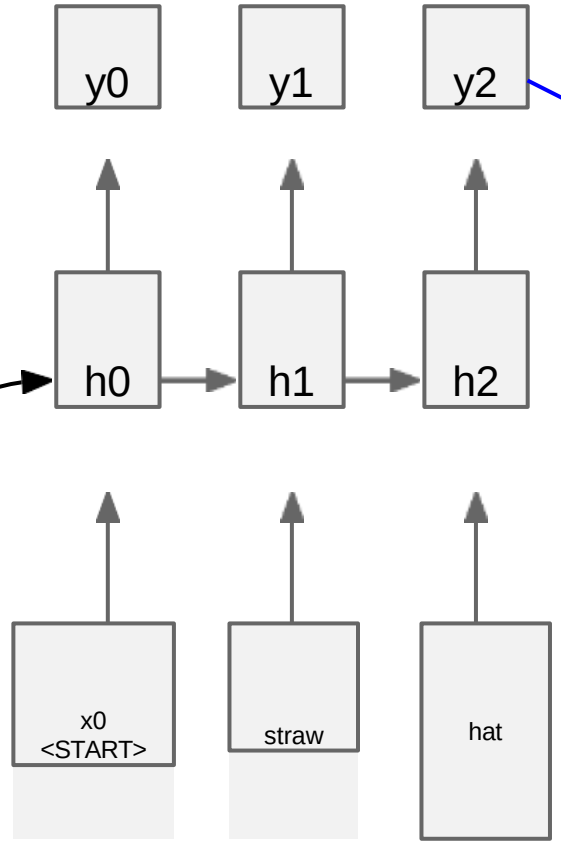
h0 h1 h2

x0
<START>

straw

hat

测试图像



采样
<END> 标志
=> 完成

图像描述: 实例

Captions generated using neuraltalk2 All images are CC0 Public domain: cat suitcase, cat tree, dog, bear, surfers, tennis, giraffe, motorcycle

A cat sitting on a suitcase on the floor

A cat is sitting on a tree branch

A dog is running in the grass with a frisbee

A white teddy bear sitting in the grass

Two people walking on the beach with surfboards

A tennis player in action on the court

Two giraffes standing in a grassy field

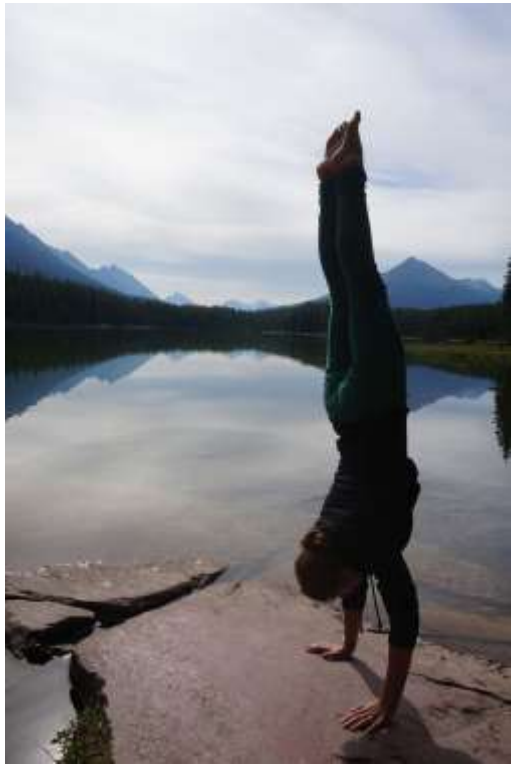
A man riding a dirt bike on a dirt track

图像描述: 失败案例

Captions generated using neuraltalk2 All images are CC0 Public domain: fur coat, handstand, spider web, baseball

A woman is holding a cat in her hand

A person holding a computer mouse on a desk

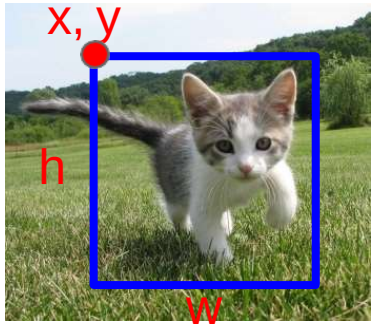


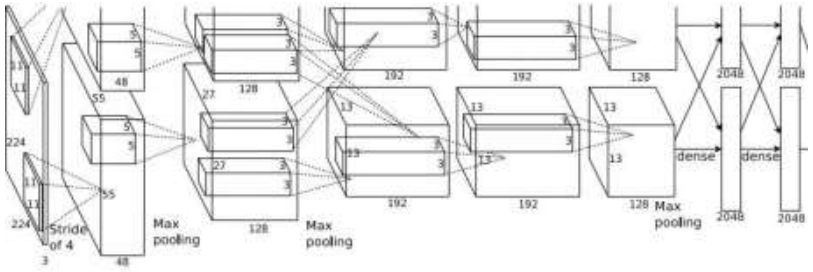
A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

A man in a baseball uniform throwing a ball

目标检测：单个对象
（分类+定位）





Vector: 4096

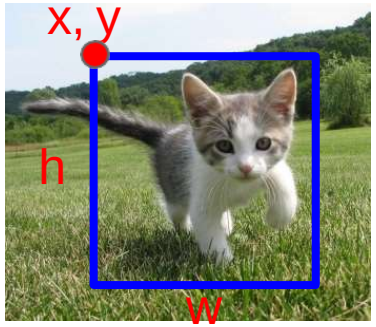
Fully Connected: 4096 to 1000

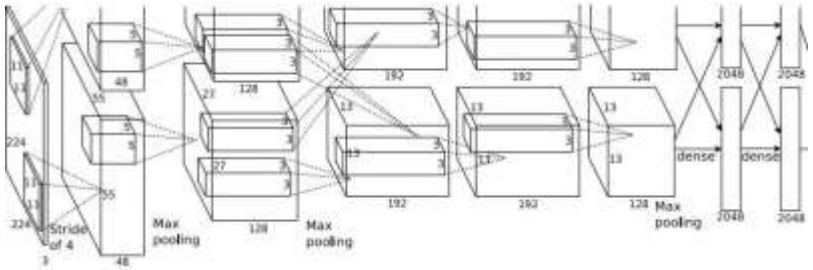
Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

目标检测：单个对象
（分类+定位）





Correct label:
Cat

Fully Connected:
4096 to 1000

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Softmax Loss

Vector:
4096

Fully Connected:
4096 to 4

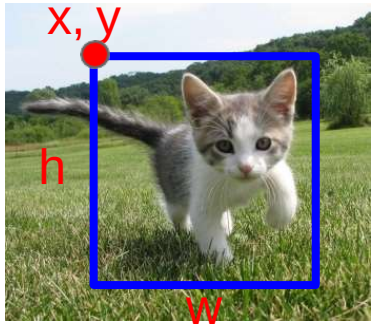
Box Coordinates
(x, y, w, h)

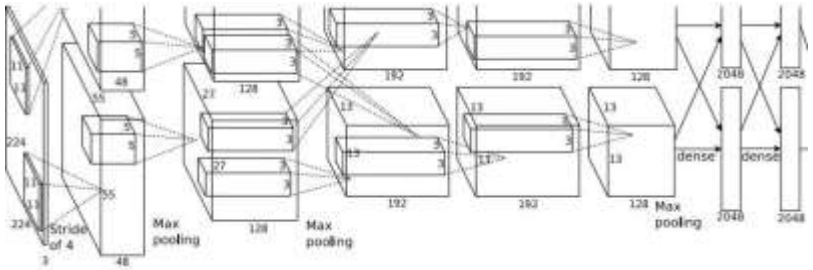
L2 Loss

Correct box:
(x', y', w', h')

将定位视为一个回归问题！

目标检测：单个对象
（分类+定位）





Correct label:
Cat

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Softmax Loss

Fully Connected:
4096 to 1000

多任务损失

Loss

Vector:
4096

Fully Connected:
4096 to 4

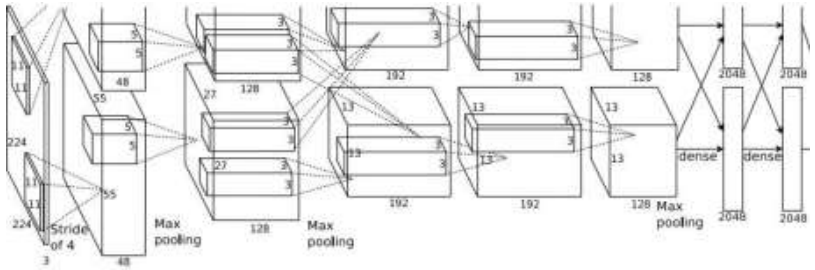
Box Coordinates
(x, y, w, h)

L2 Loss

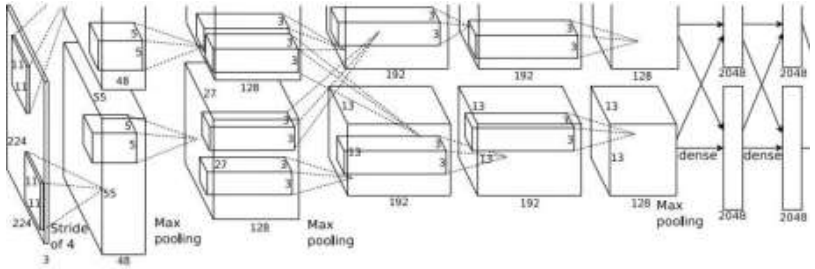
Correct box:
(x', y', w', h')

将定位视为一个回归问题！

目标检测：多个对象

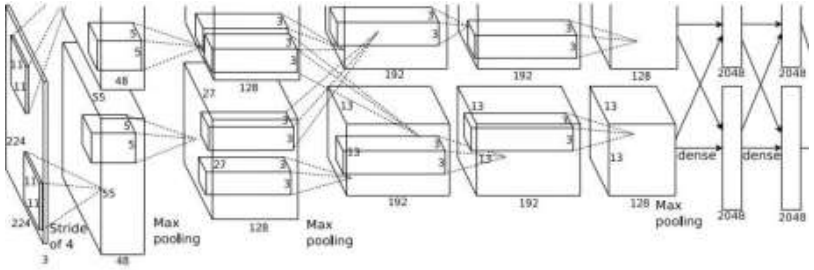


CAT: (x, y, w, h)



DOG: (x, y, w, h)

DOG: (x, y, w, h)



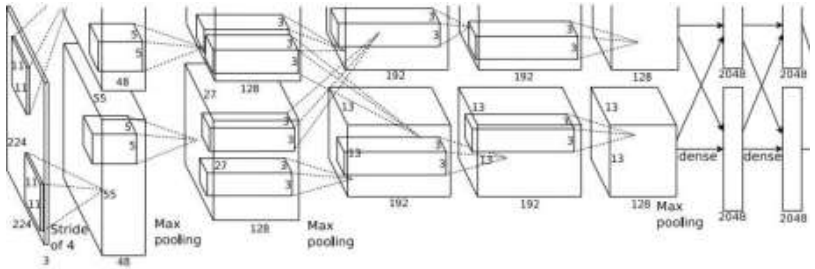
CAT: (x, y, w, h)

DUCK: (x, y, w, h)

DUCK: (x, y, w, h)

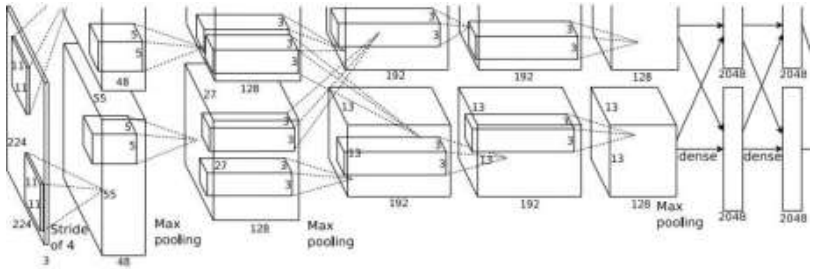
目标检测：多个对象

每张图片有不同数量的输出



CAT: (x, y, w, h)

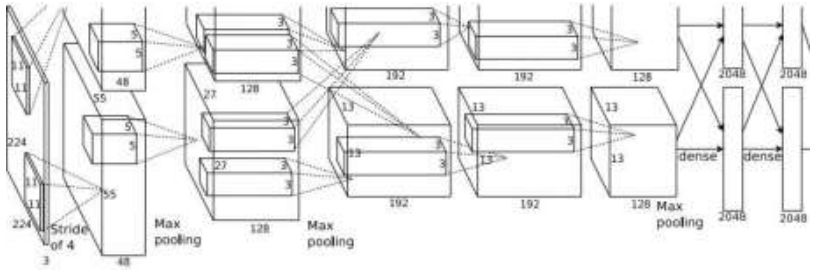
4个数字



DOG: (x, y, w, h)

DOG: (x, y, w, h)

12个数字



CAT: (x, y, w, h)

DUCK: (x, y, w, h)

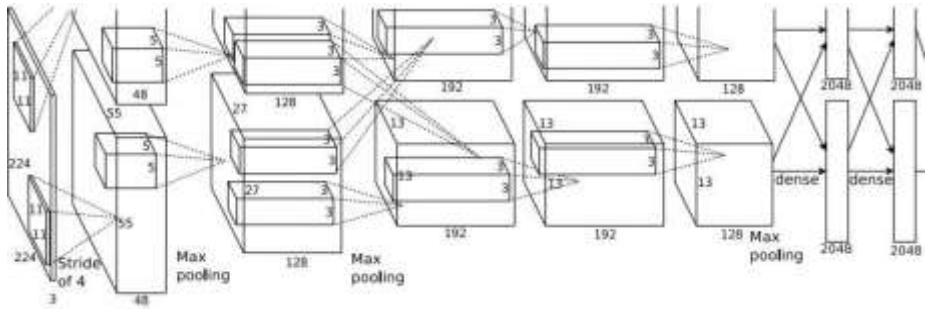
DUCK: (x, y, w, h)

….

很多数字！

目标检测：多个对象

利用CNN将图片的许多裁剪部分(crop)分类为对象或背景



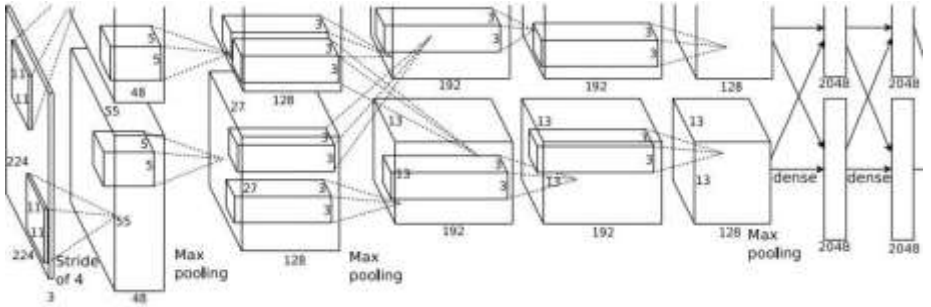
Dog? NO

Cat? NO

Background? YES

目标检测：多个对象

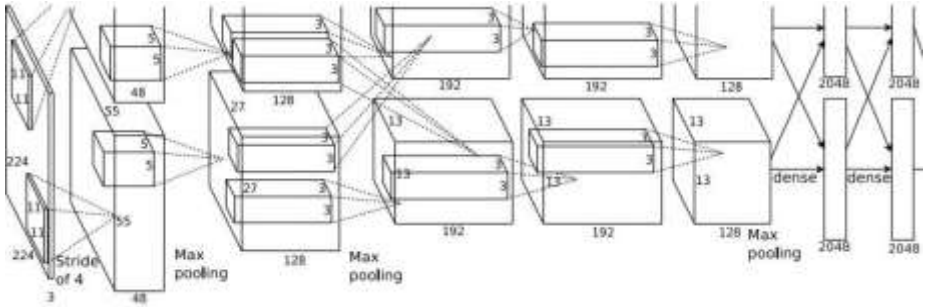
利用CNN将图片的许多裁剪部分(crop)分类为对象或背景



Dog? YES
Cat? NO
Background? NO

目标检测：多个对象

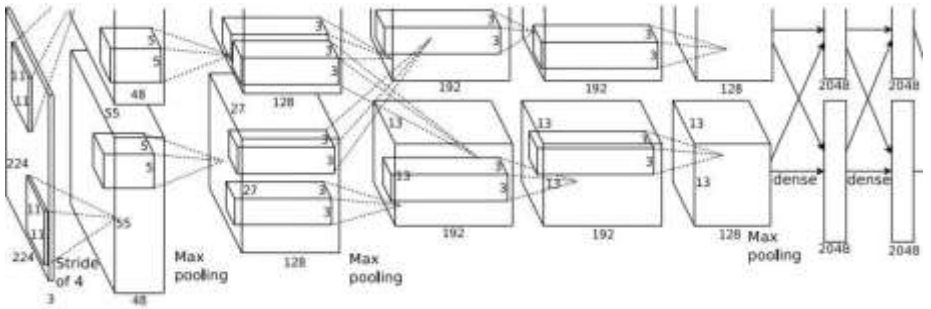
利用CNN将图片的许多裁剪部分(crop)分类为对象或背景



Dog? YES
Cat? NO
Background? NO

目标检测：多个对象

利用CNN将图片的许多裁剪部分(crop)分类为对象或背景

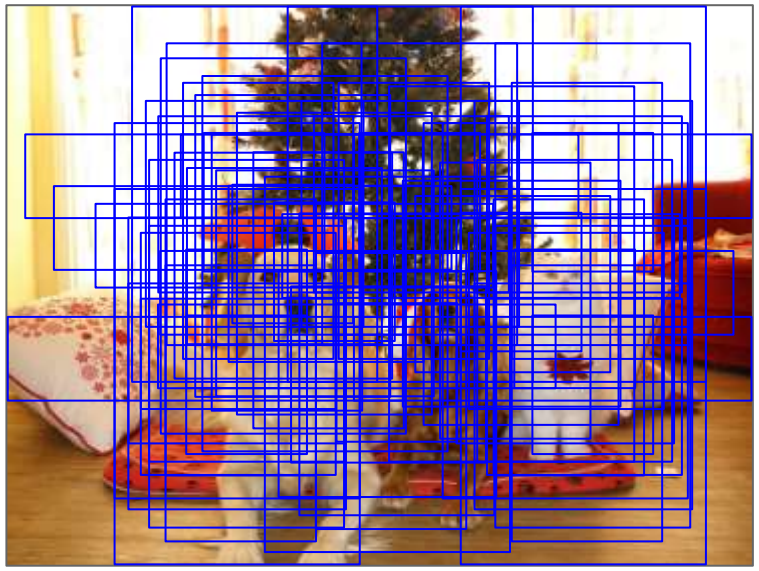


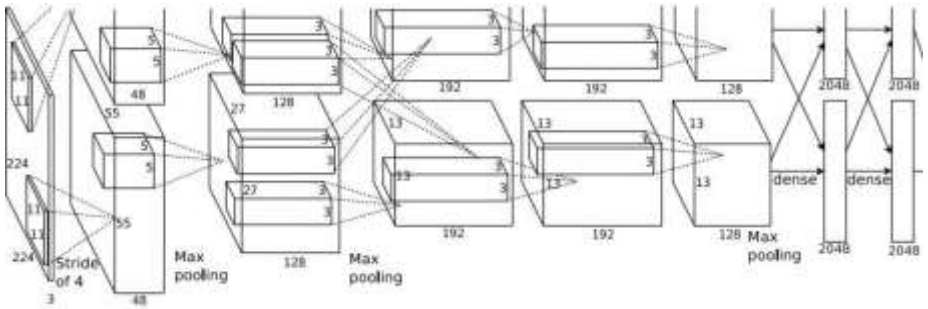
Dog? NO
Cat? YES
Background? NO

问：这种方法有什么问题？

目标检测：多个对象

利用CNN将图片的许多裁剪部分(crop)分类为对象或背景



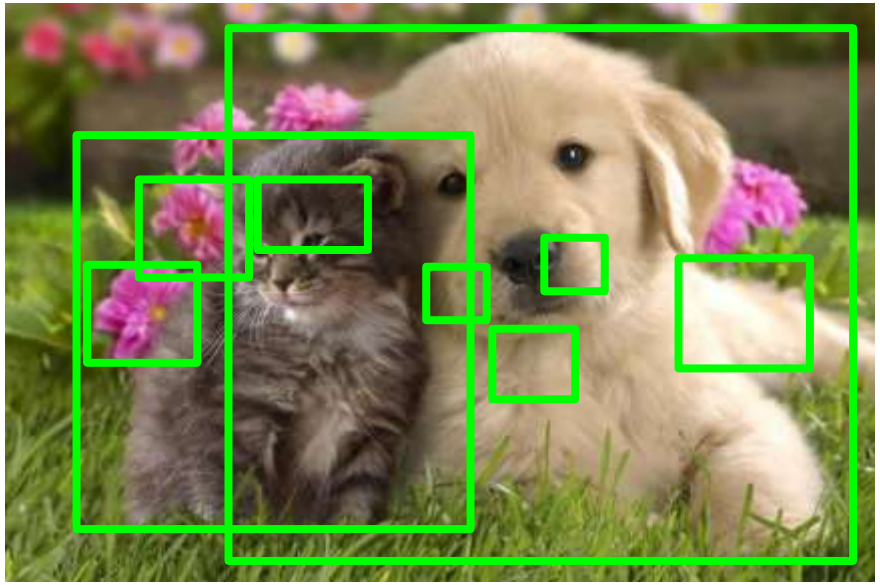


Dog? NO
Cat? YES
Background? NO

问题：需要用CNN处理大量的位置、尺度和纵横比等信息，计算代价很大！

候选区域：选择性搜索(Selective Search)

- 寻找可能包含目标对象的图片区域
- 运行相对较快；例如，在CPU上选择性搜索给出2000个候选区域只需几秒钟



Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012 Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013
Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014 Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”, ECCV 2014

R-CNN

输入图片

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

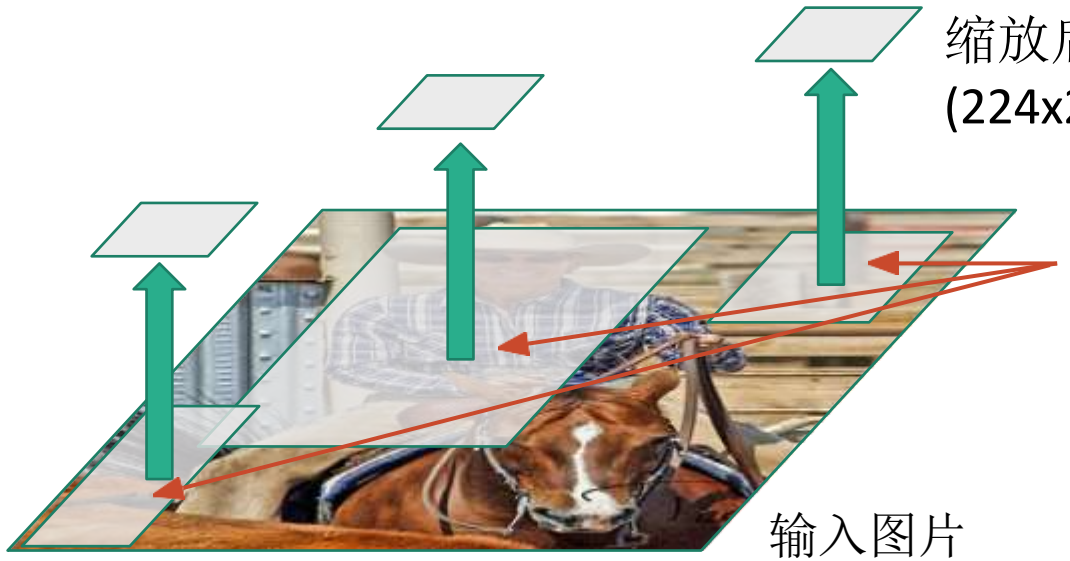
R-CNN

由推荐算法得到的感兴趣区域(~2k)

输入图片

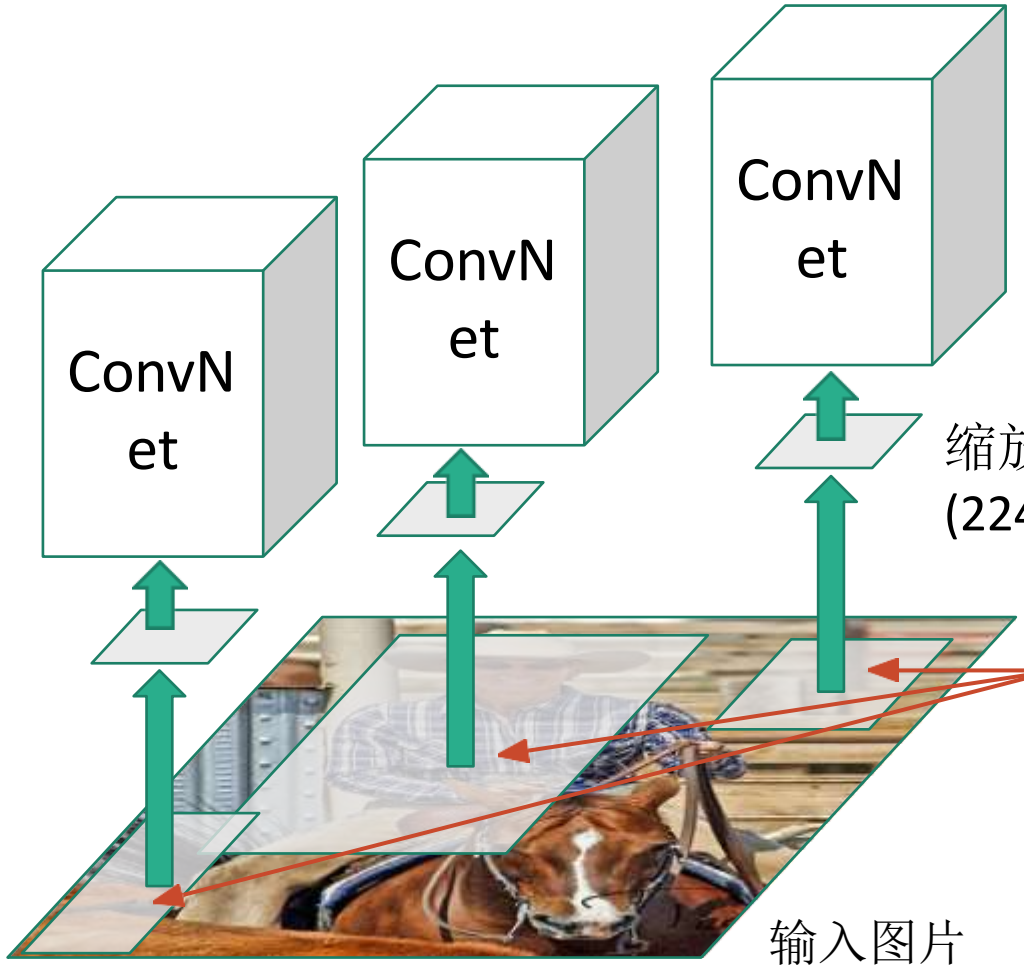
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.

R-CNN



Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN



将每个区域输入到卷积网络中前向传播（网络在ImageNet上预训练过）

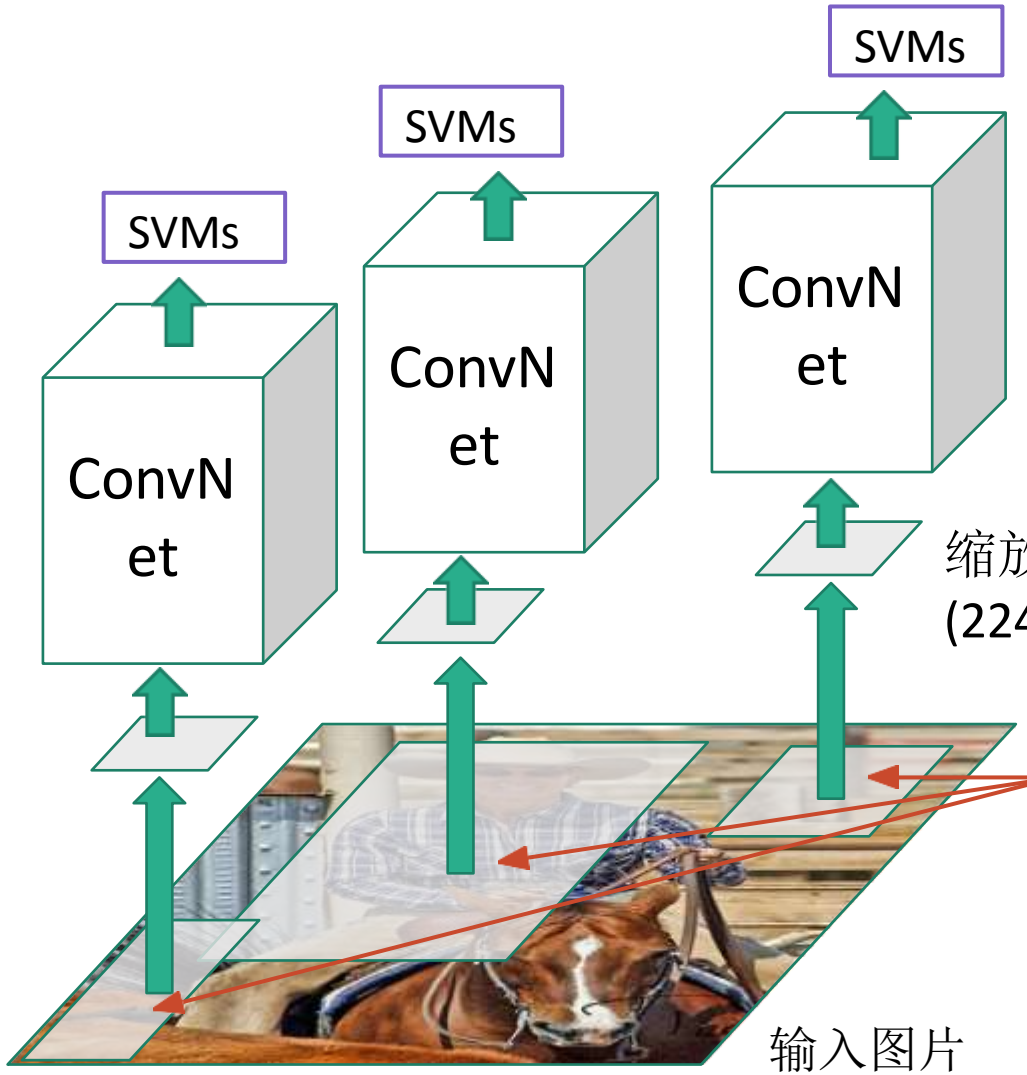
缩放后的图像区域
(224x224 像素)

由推荐算法得到的感兴趣区域(~2k)

输入图片

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

R-CNN



用支持向量机对每个区域分类

将每个区域输入到卷积网络中前向传播（网络在ImageNet上预训练过）

缩放后的图像区域
(224x224 像素)

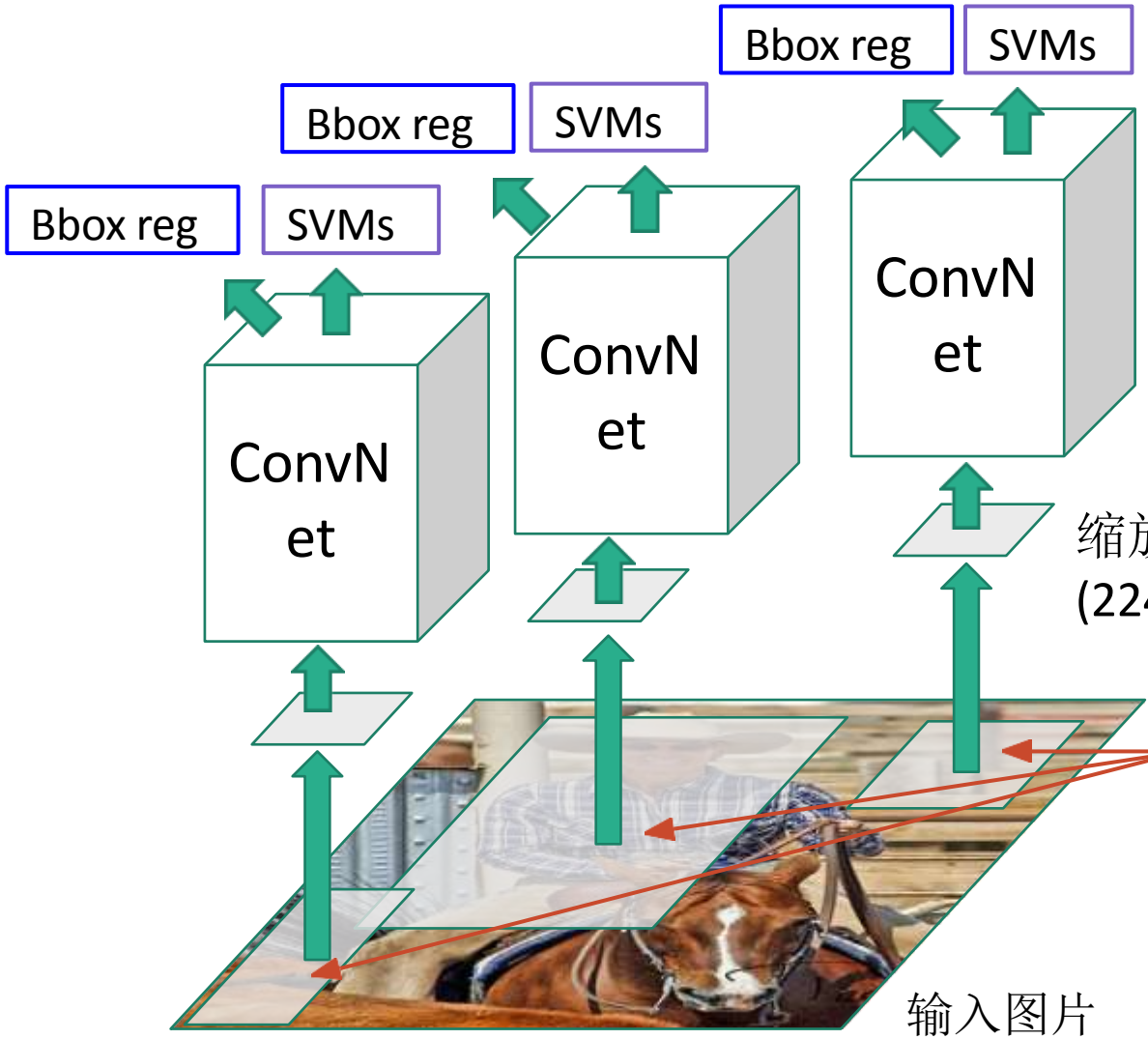
由推荐算法得到的感兴趣区域(~2k)

输入图片

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

R-CNN

对预测边框进行修正：4个数：(dx, dy, dw, dh)



用支持向量机对每个区域分类

将每个区域输入到卷积网络中前向传播（网络在ImageNet上预训练过）

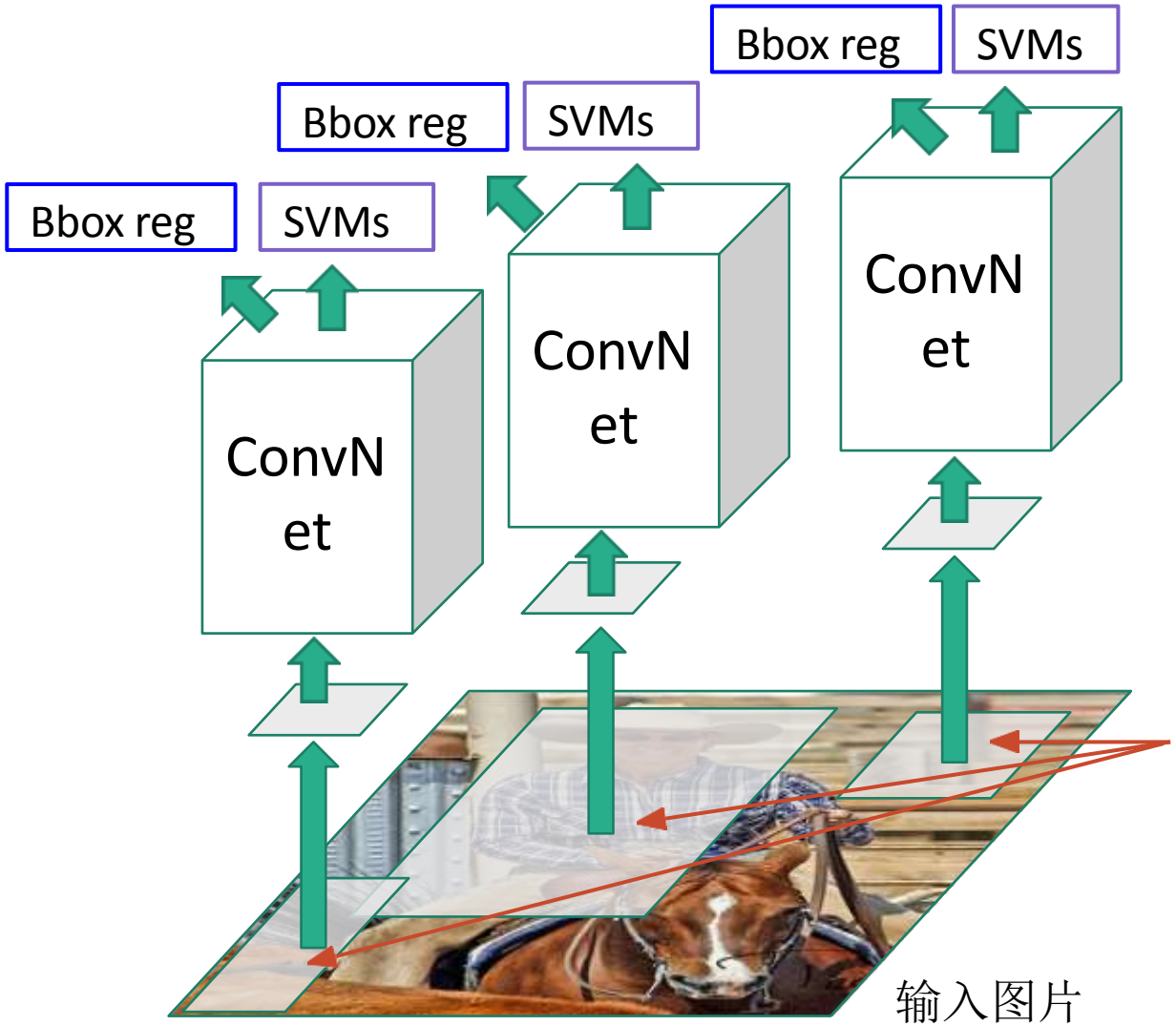
缩放后的图像区域
(224x224 像素)

由推荐算法得到的感兴趣区域(~2k)

输入图片

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

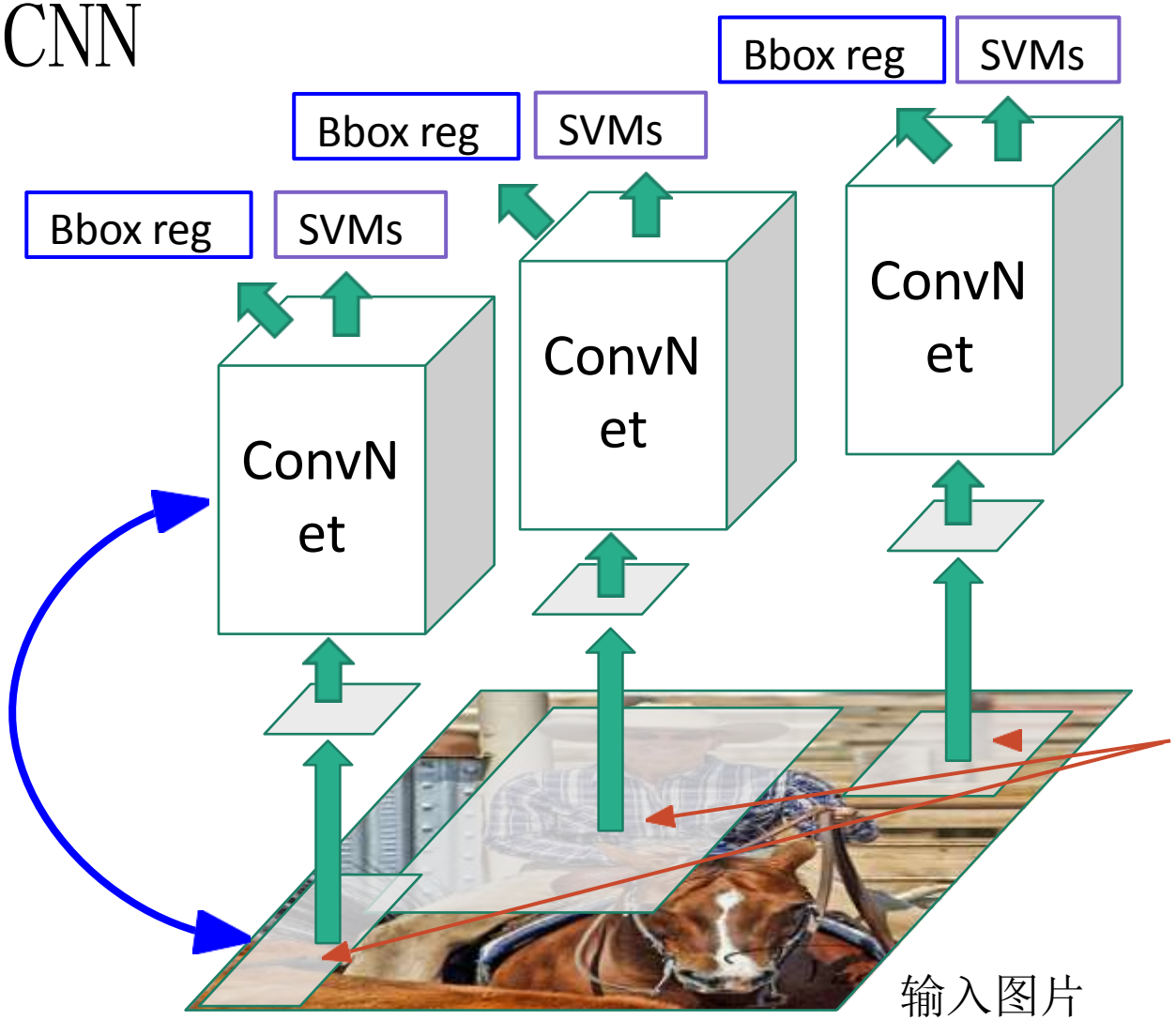
R-CNN



问题：非常慢！需要对每张图片做~2k次独立的前向传播！

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

“Slow” R-CNN



问题：非常慢！需要对每张图片做~2k次独立的前向传播！

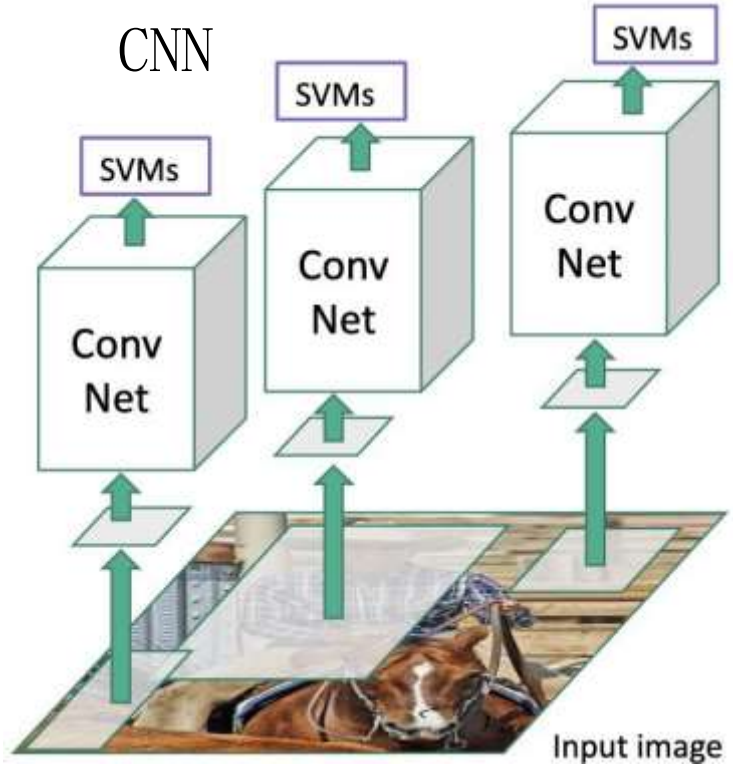
方法：先把图片输入卷积网络，再对特征进行裁剪！

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN

输入图片

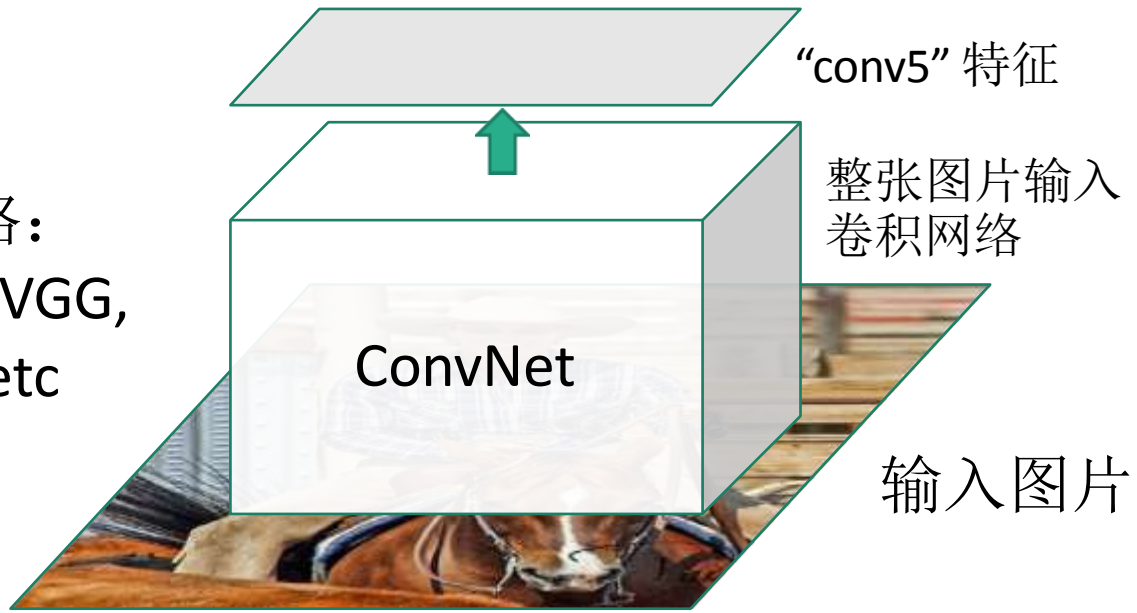
“Slow” R-CNN



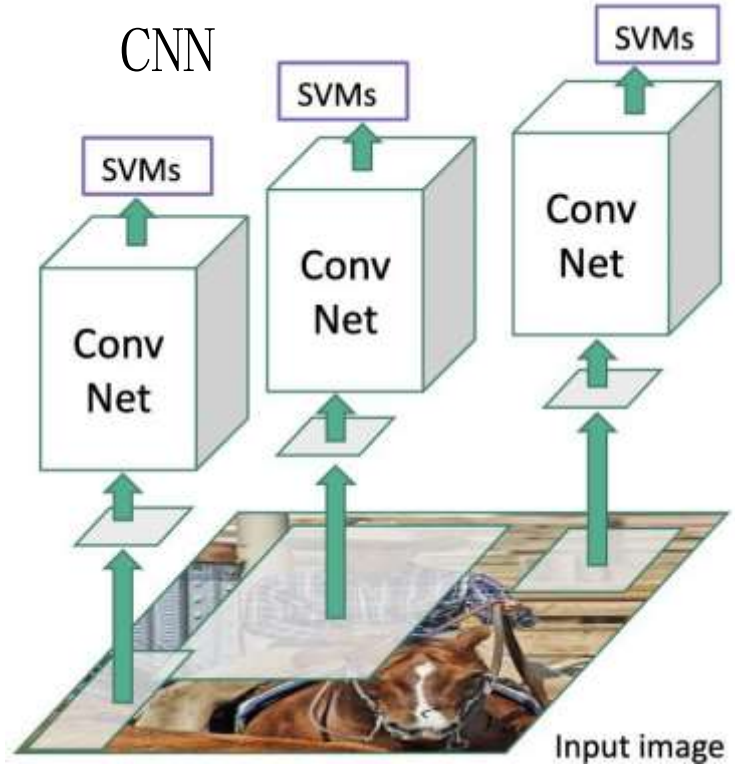
Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN

骨干网络：
AlexNet, VGG,
ResNet, etc

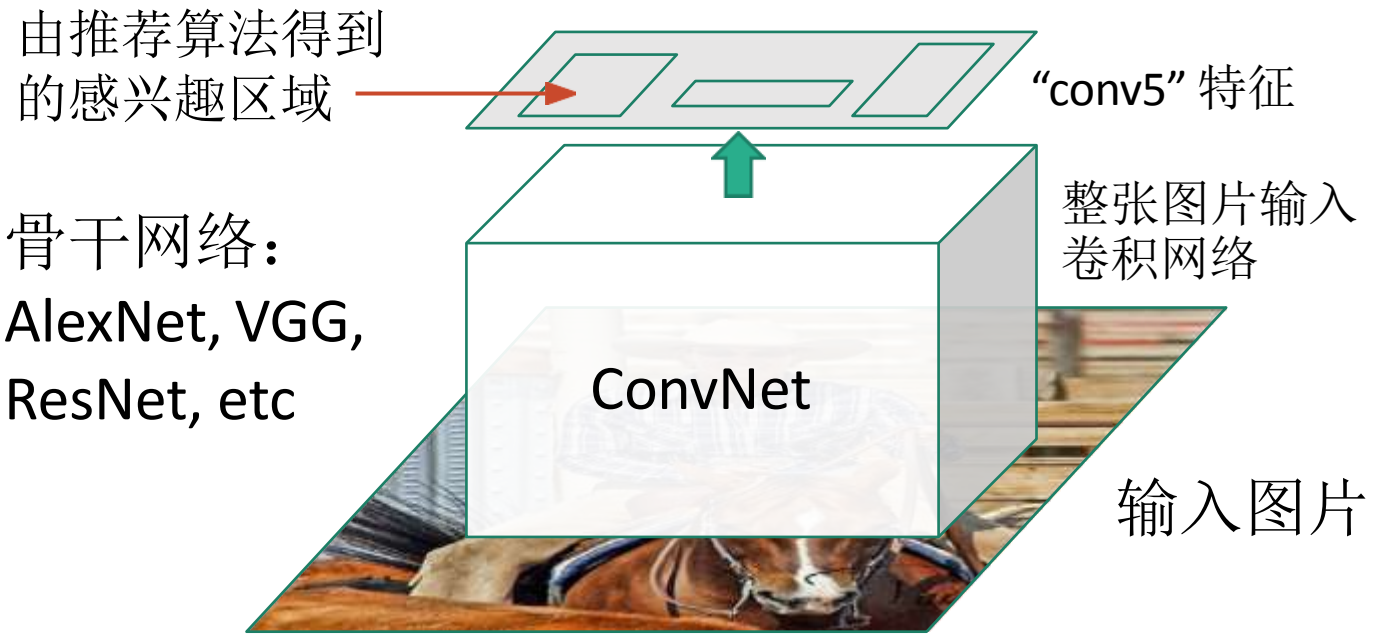


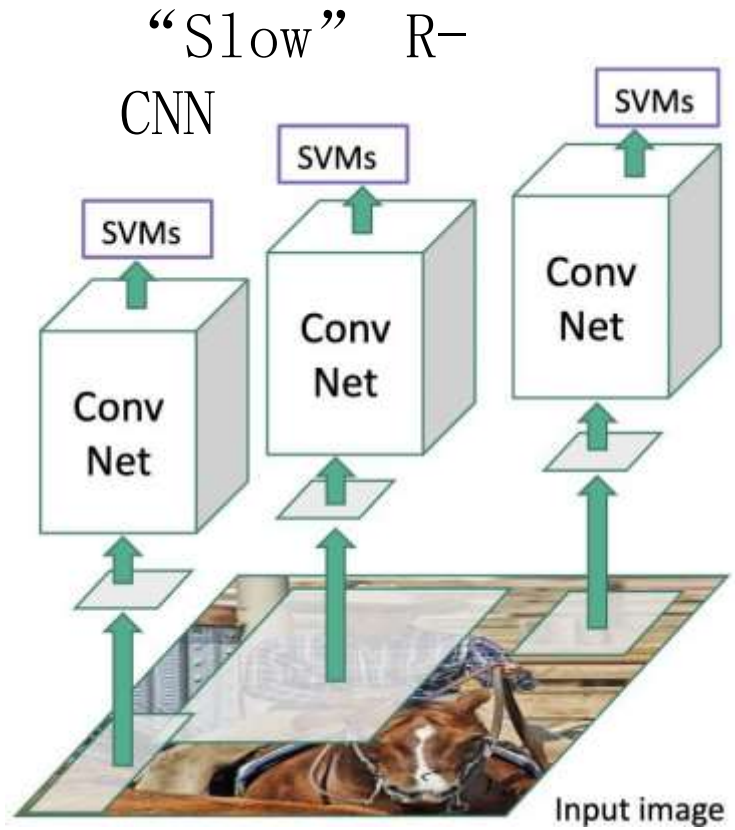
“Slow” R-CNN



Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

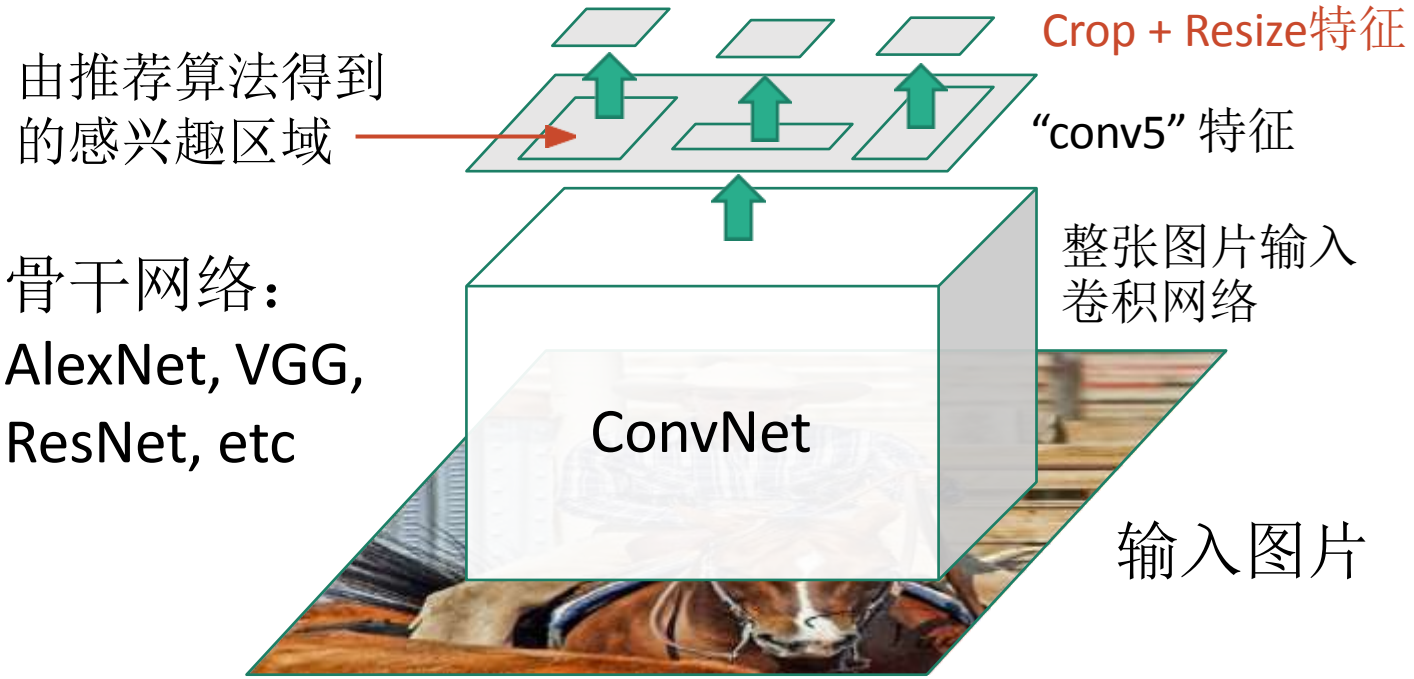
Fast R-CNN

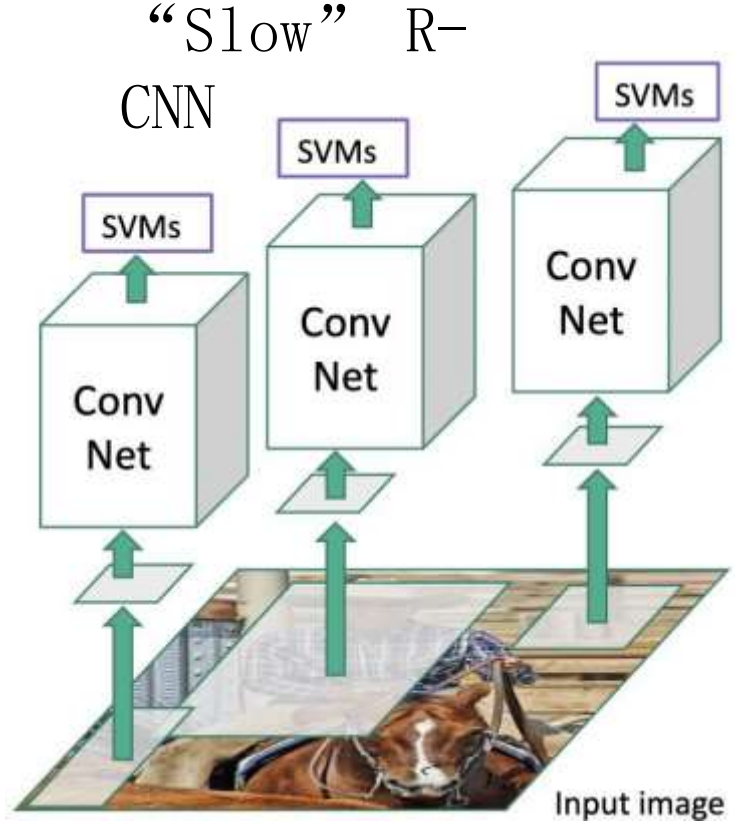




Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

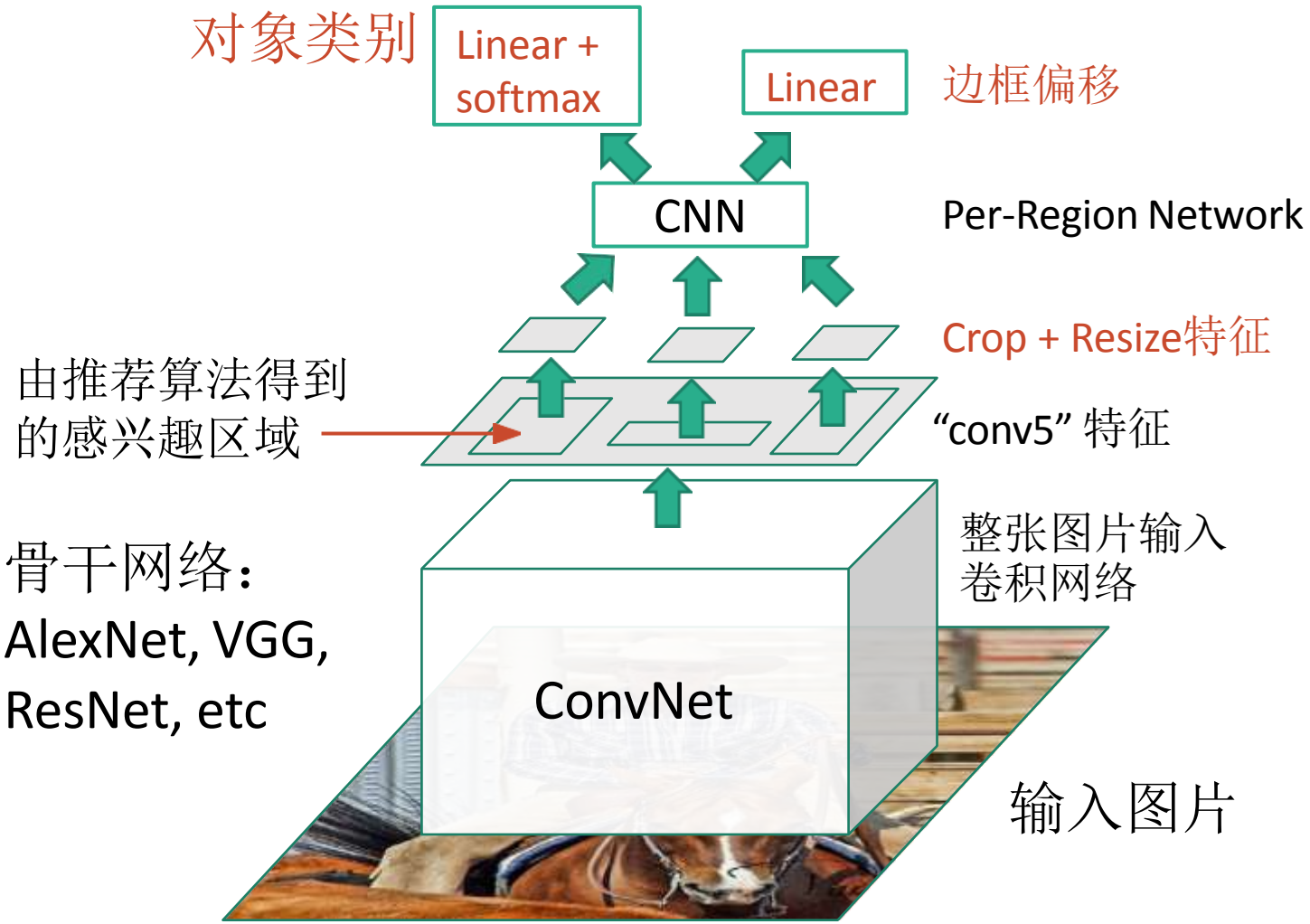
Fast R-CNN

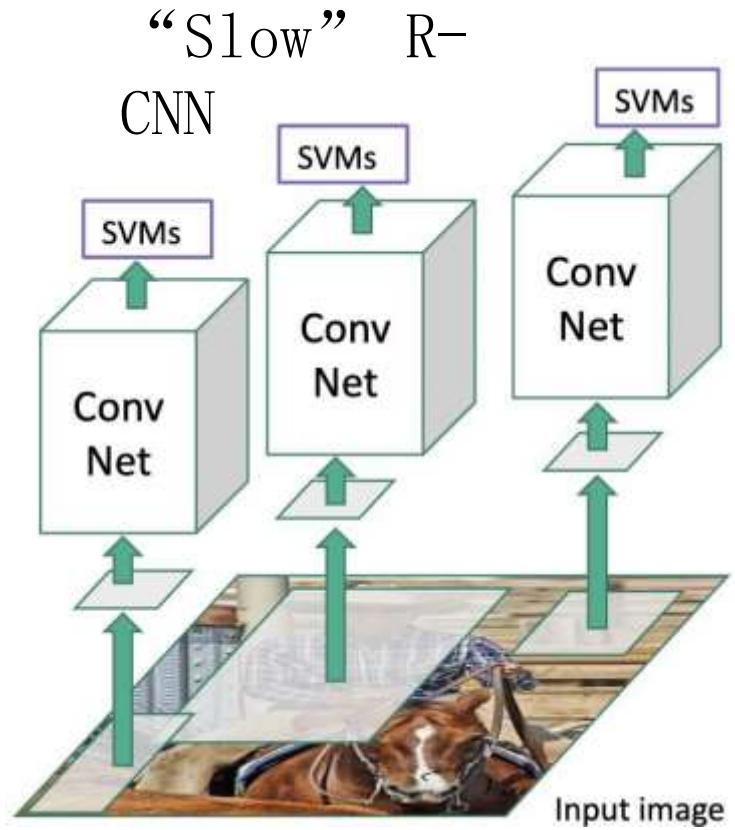




Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

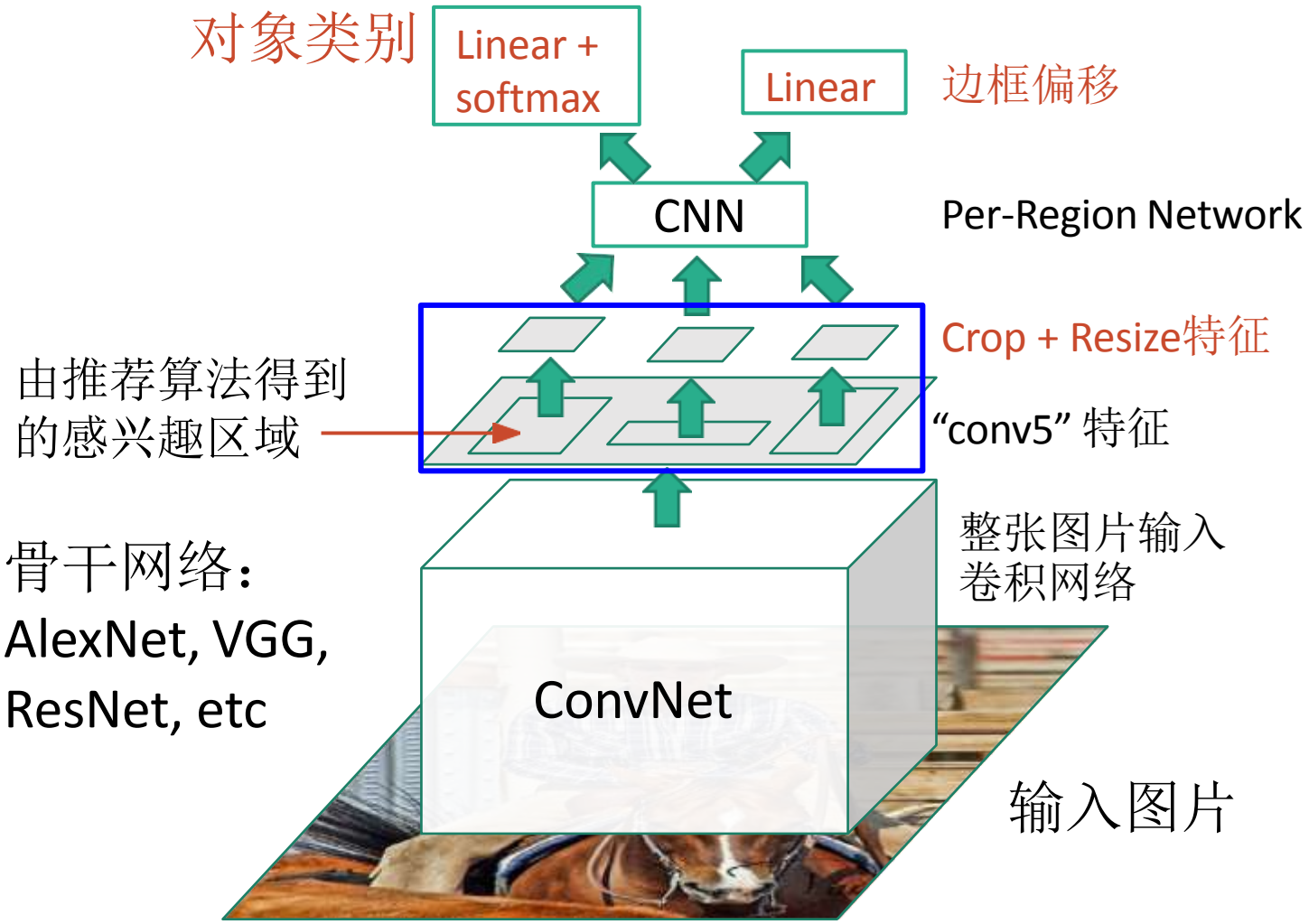
Fast R-CNN

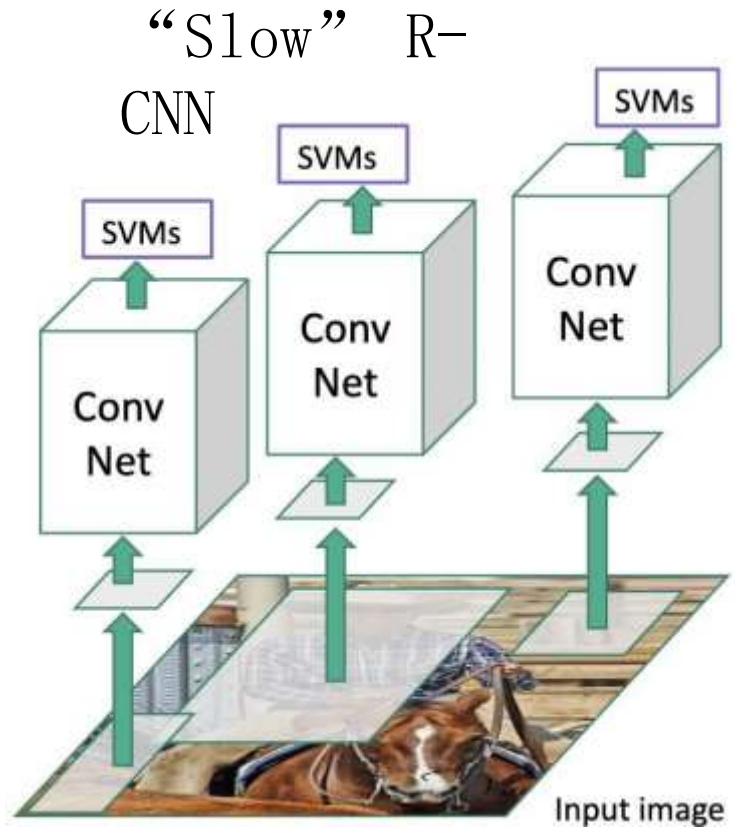




Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

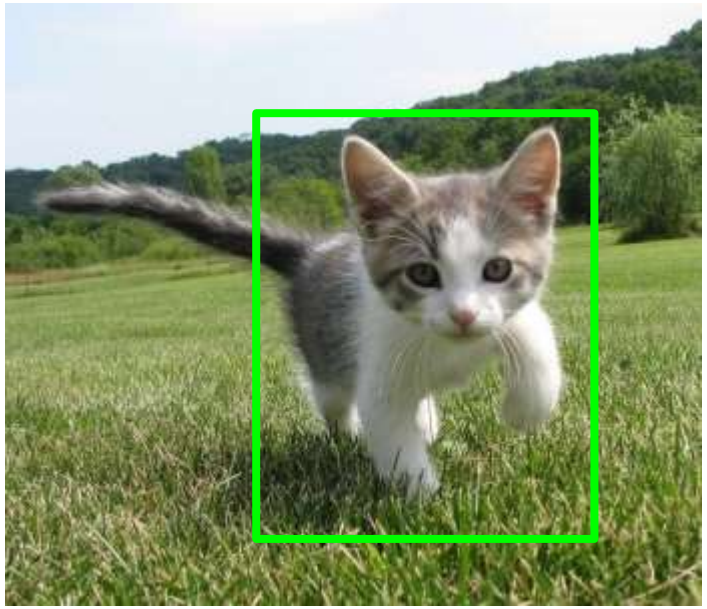
Fast R-CNN





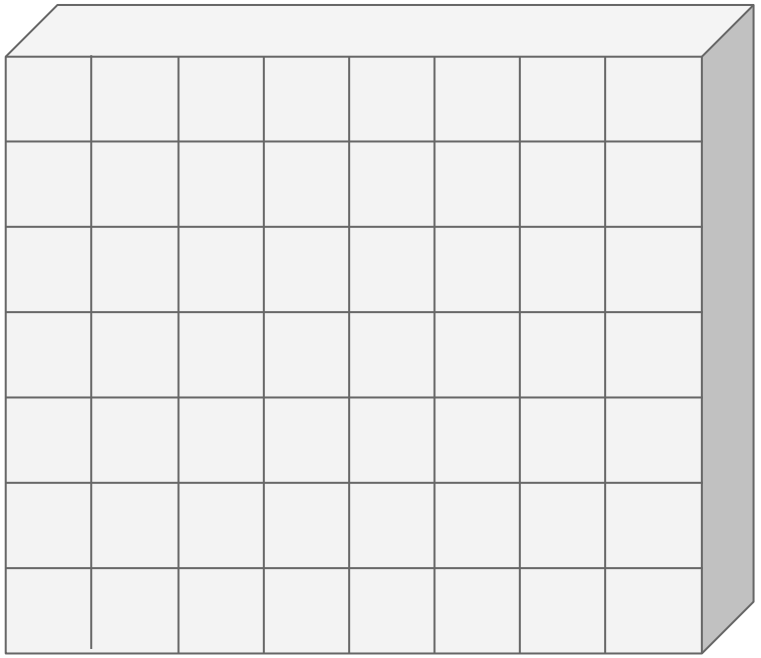
Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

裁剪特征：RoI池化



输入图片
(e.g. 3 x 640 x 480)

CNN



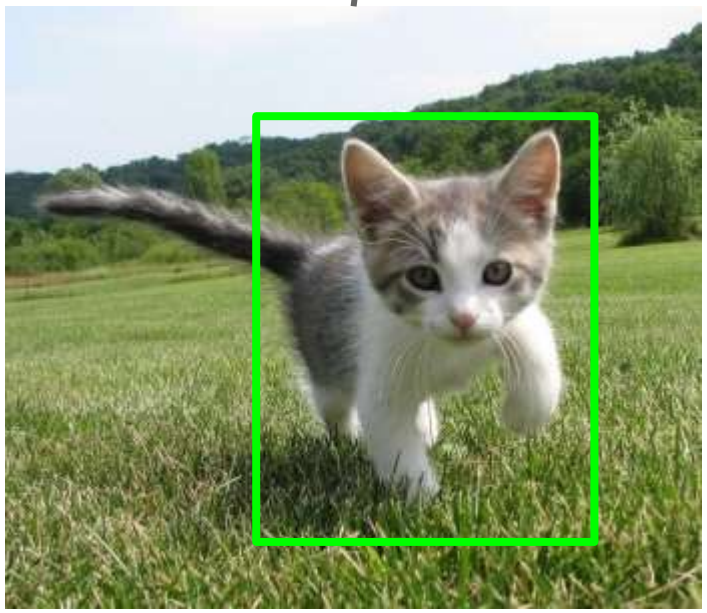
图片特征
(e.g. 512 x 20 x 15)

Girshick, “Fast R-CNN”, ICCV 2015.

Girshick, “Fast R-CNN”, ICCV 2015.

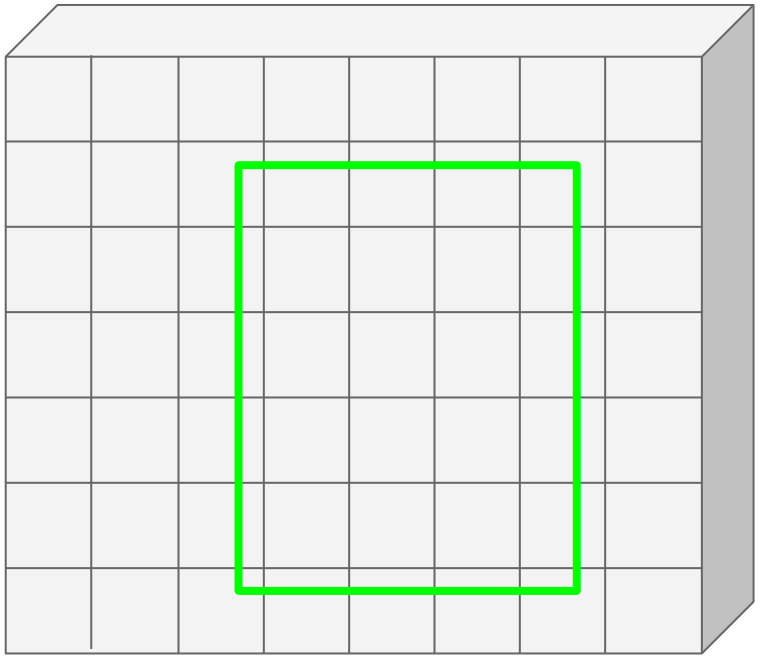
裁剪特征：RoI池化

候选区域映射到特征上



输入图片
(e.g. 3 x 640 x 480)

CNN



图片特征
(e.g. 512 x 20 x 15)

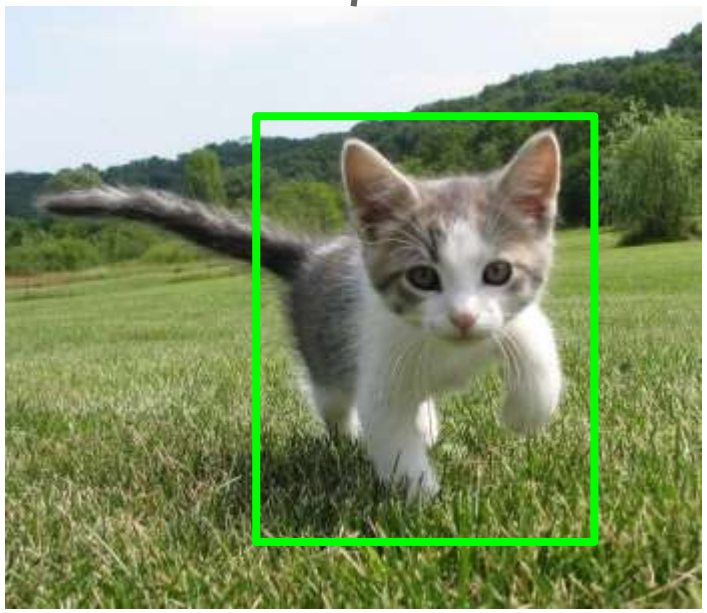
Girshick, “Fast R-CNN”, ICCV 2015.

Girshick, “Fast R-CNN”, ICCV 2015.

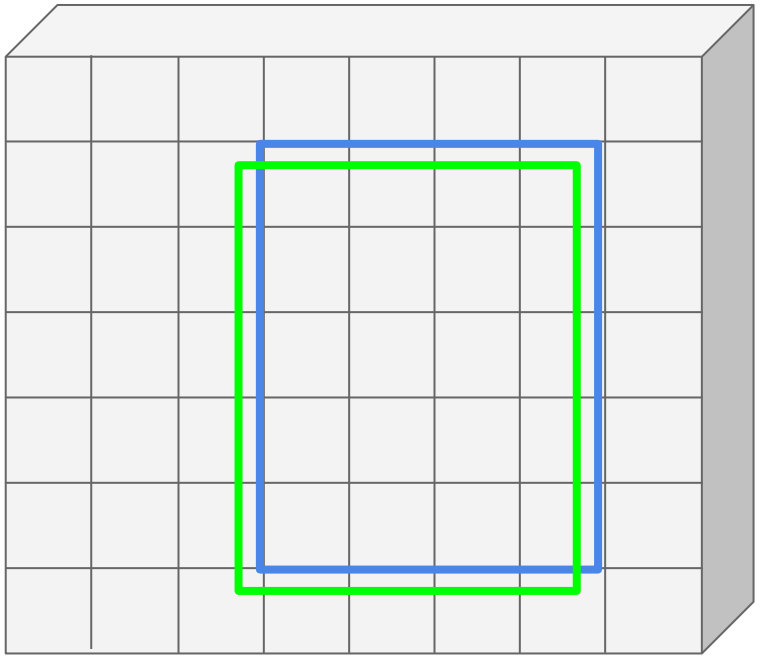
裁剪特征：RoI池化

候选区域映射到特征上

“对齐”到网格单元



CNN



输入图片
(e.g. 3 x 640 x 480)

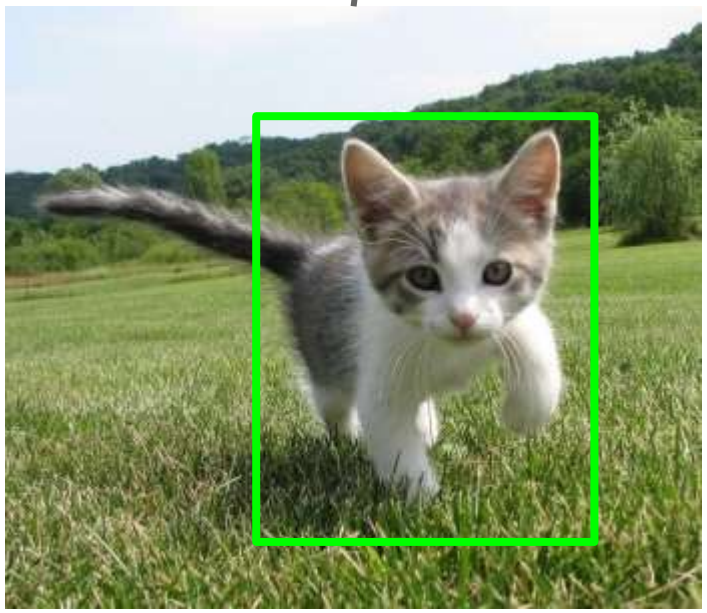
图片特征
(e.g. 512 x 20 x 15)

Girshick, “Fast R-CNN”, ICCV 2015.

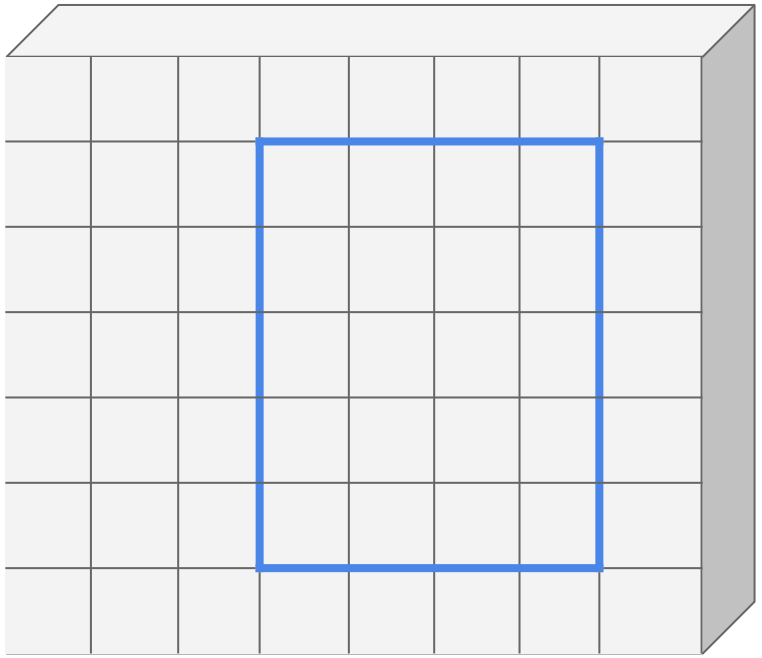
裁剪特征：RoI池化

候选区域映射到特征上

“对齐”到网格单元



CNN



输入图片
(e.g. 3 x 640 x 480)

图片特征
(e.g. 512 x 20 x 15)

问：我们如何把512 x 20 x 15的张量调整为512 x 2 x 2 大小？

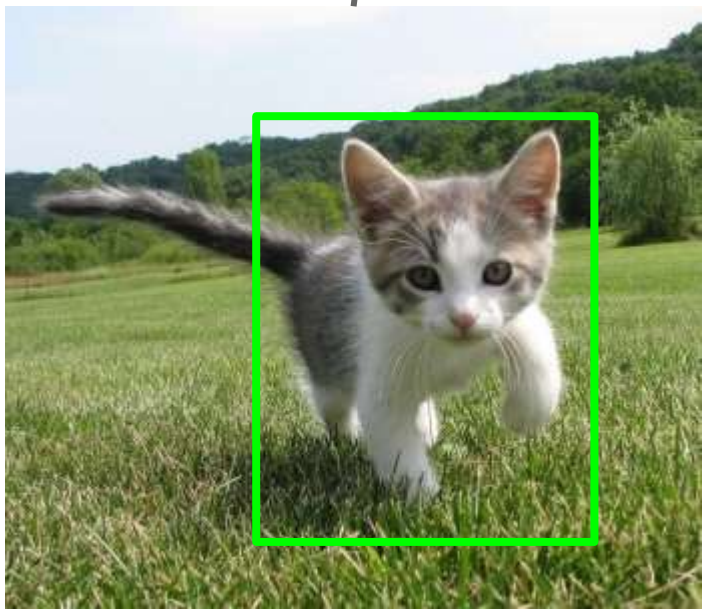
Girshick, “Fast R-CNN”, ICCV 2015.

裁剪特征：RoI池化

候选区域映射到特征上

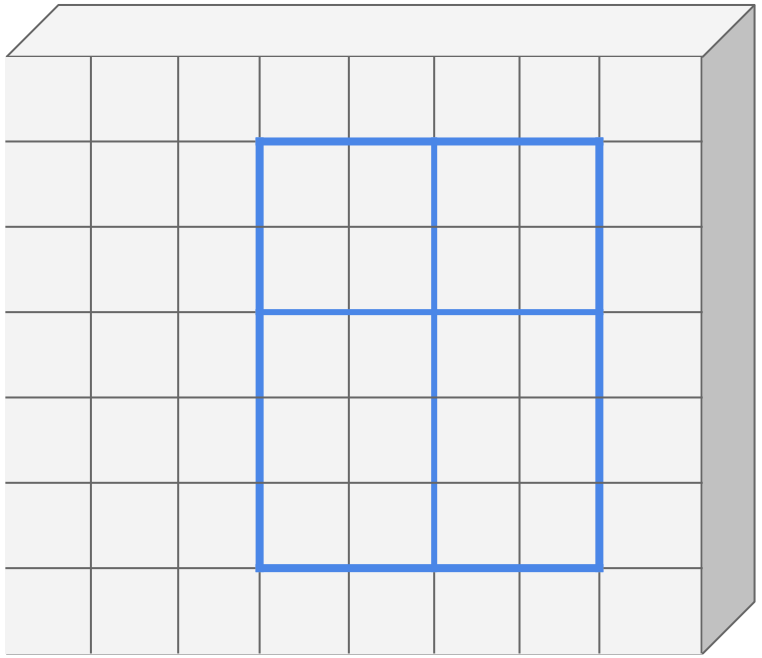
“对齐”到网格单元

粗略地分成2x2同等大小的子区域



输入图片
(e.g. 3 x 640 x 480)

CNN



图片特征
(e.g. 512 x 20 x 15)

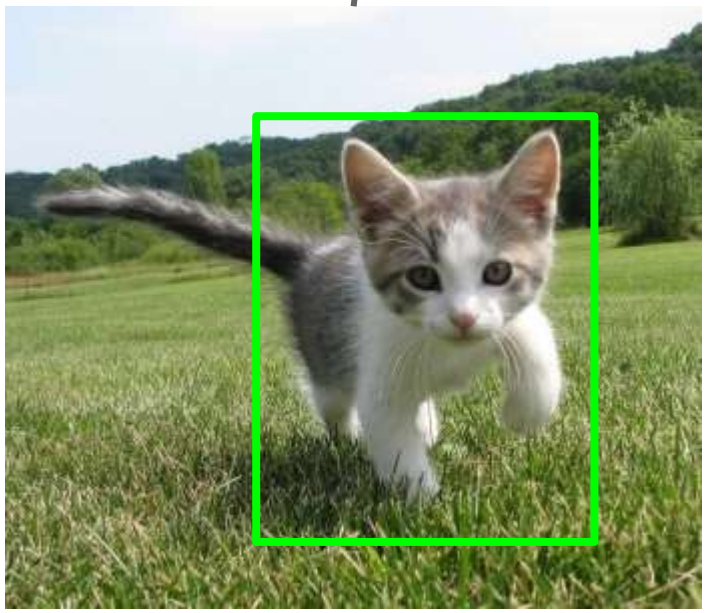
问：我们如何把512 x 20 x 15的张量调整为512 x 2 x 2 大小？

Girshick, “Fast R-CNN”, ICCV 2015.

裁剪特征：RoI池化

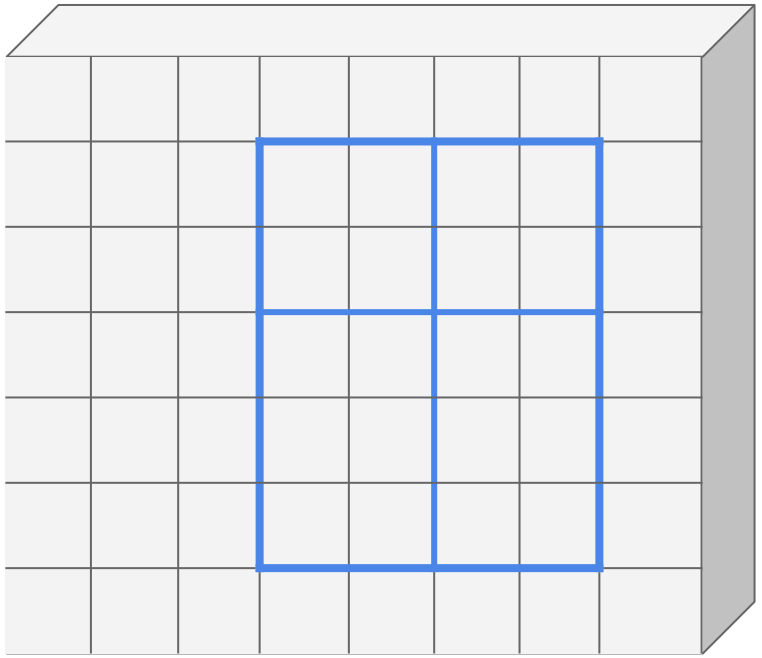
候选区域映射到特征上

“对齐”到网格单元



输入图片
(e.g. 3 x 640 x 480)

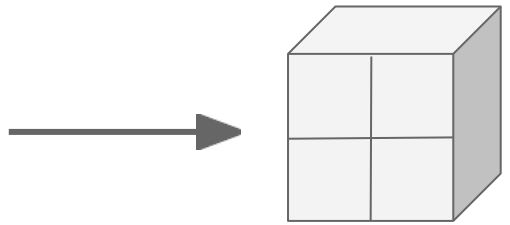
CNN



图片特征
(e.g. 512 x 20 x 15)

粗略地分成2x2同等大小的子区域

在每个子区域做最大池化



区域特征(这里是512 x 2 x 2;
实际操作中可能是512 x 7 x 7)

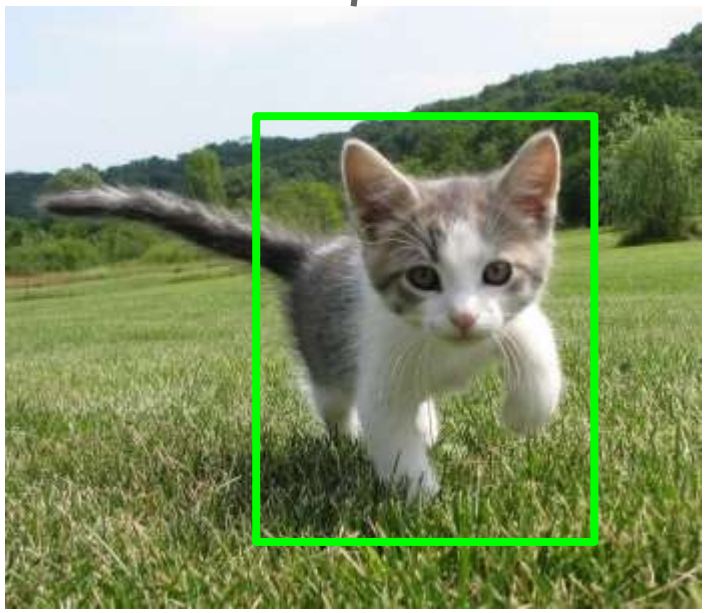
即使原始区域尺寸不同，区域特征的尺寸总是相同的！

Girshick, “Fast R-CNN”, ICCV 2015.

裁剪特征：RoI池化

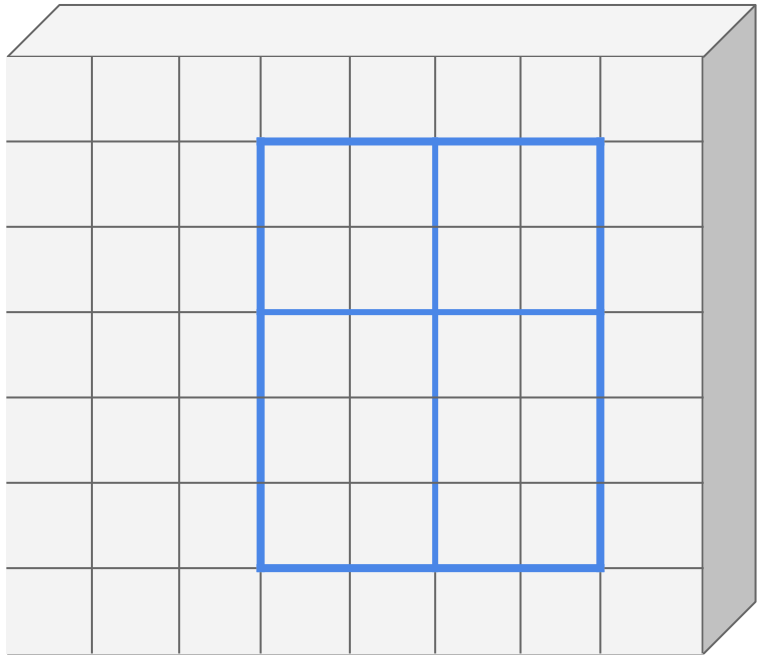
候选区域映射到特征上

“对齐”到网格单元



输入图片
(e.g. 3 x 640 x 480)

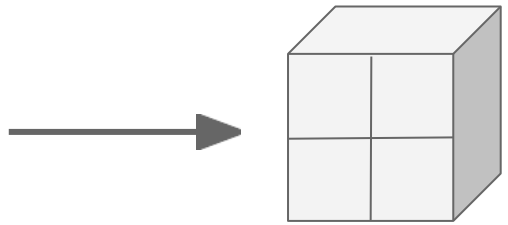
CNN



图片特征
(e.g. 512 x 20 x 15)

粗略地分成2x2同等大小的子区域

在每个子区域做最大池化



区域特征(这里是512 x 2 x 2;实际操作中可能是512 x 7 x 7)

即使原始区域尺寸不同，区域特征的尺寸总是相同的！

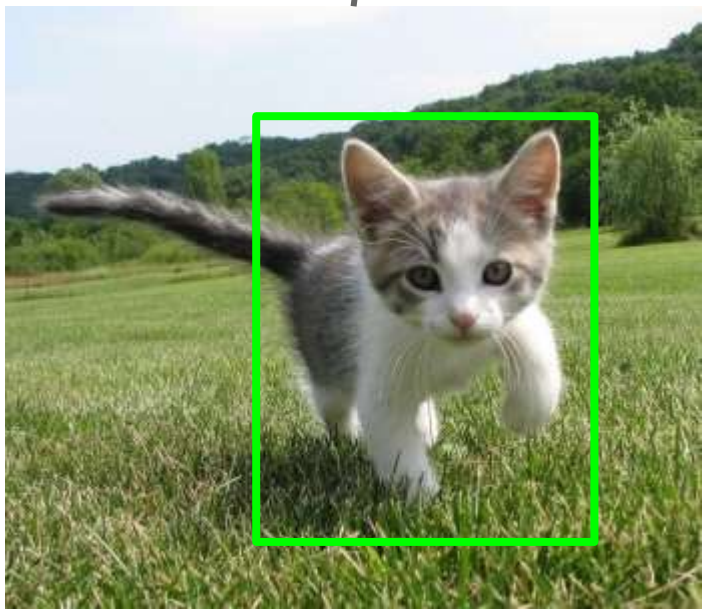
问题： 区域特征有轻微的不对齐

Girshick, “Fast R-CNN”, ICCV 2015.

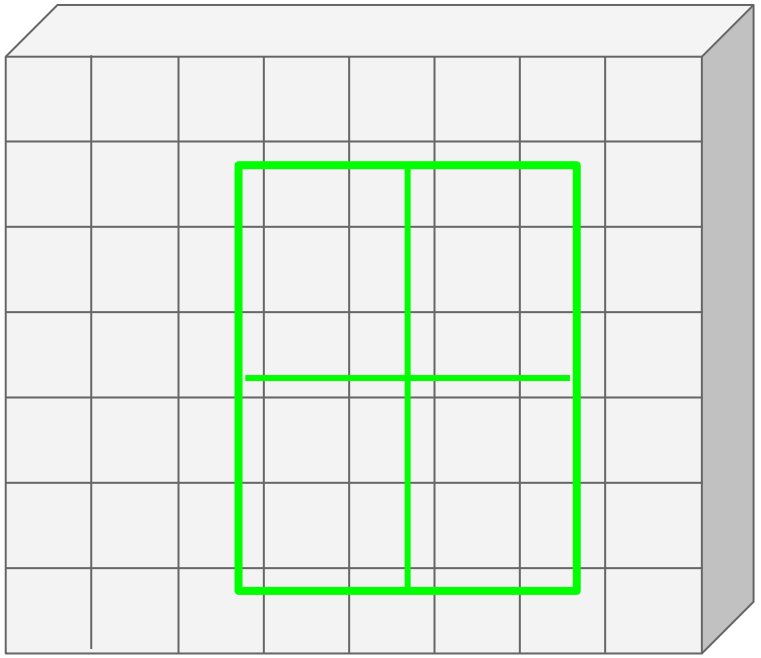
裁剪特征：RoI对齐

候选区域映射到特征上

没有“对齐”！



CNN



输入图片
(e.g. 3 x 640 x 480)

图片特征
(e.g. 512 x 20 x 15)

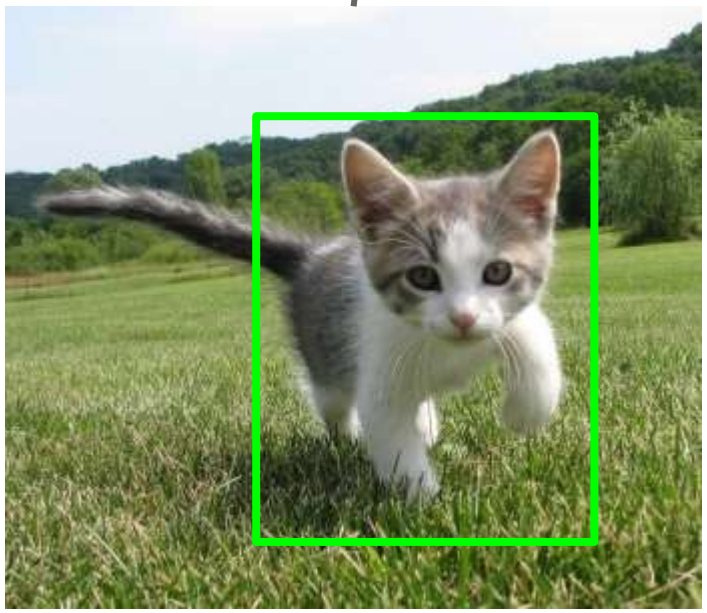
He et al, “Mask R-CNN”, ICCV 2017

裁剪特征：RoI对齐

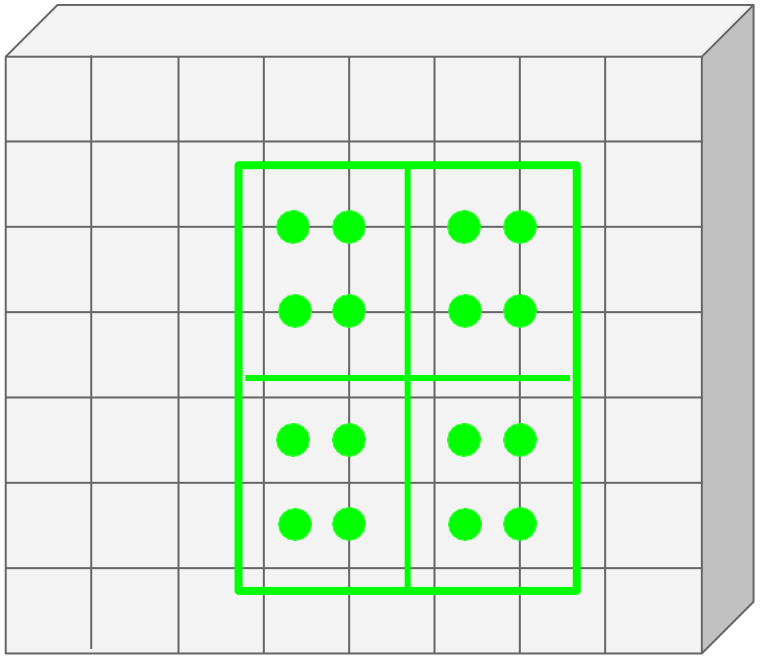
候选区域映射到特征上

没有“对齐”！

在子区域中用双线性插值法采样



CNN



输入图片
(e.g. 3 x 640 x 480)

图片特征
(e.g. 512 x 20 x 15)

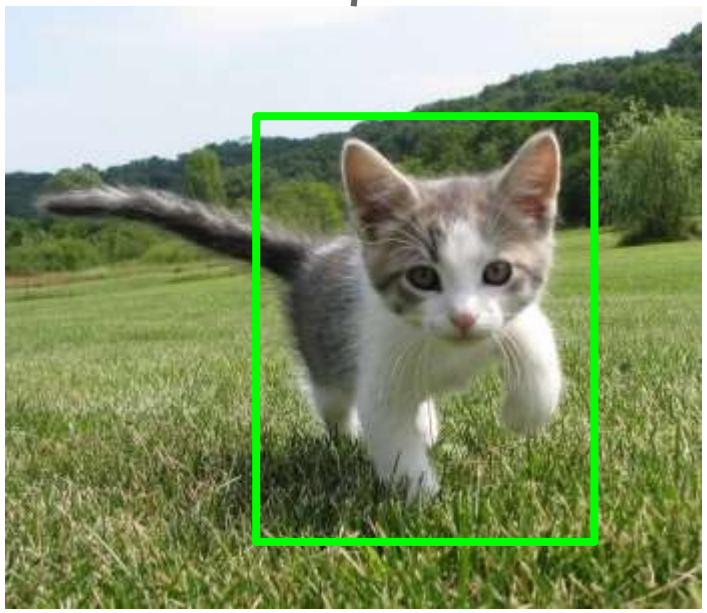
He et al, “Mask R-CNN”, ICCV 2017

裁剪特征：RoI对齐

候选区域映射到特征上

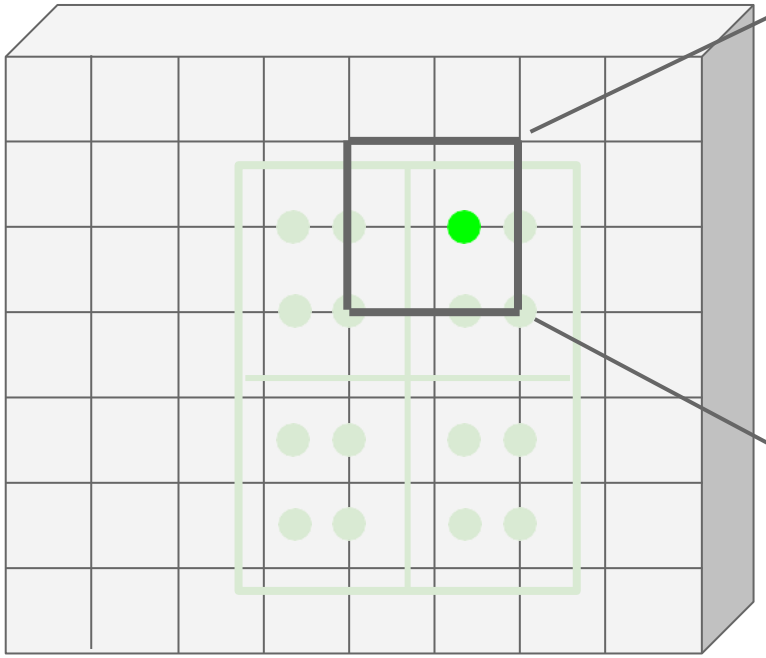
没有“对齐”！

在子区域中用双线性插值法采样

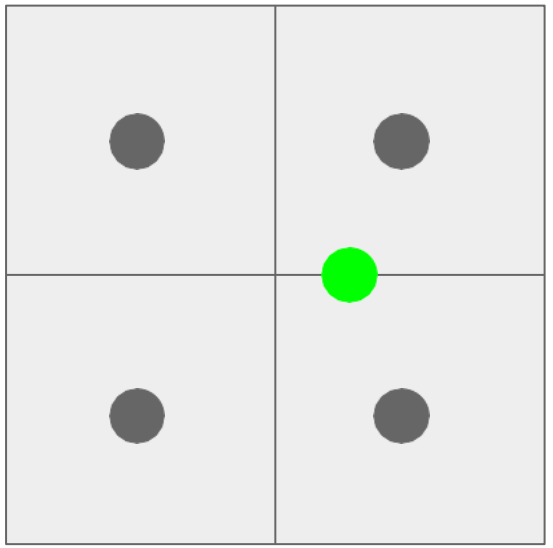


输入图片
(e.g. 3 x 640 x 480)

CNN



图片特征
(e.g. 512 x 20 x 15)



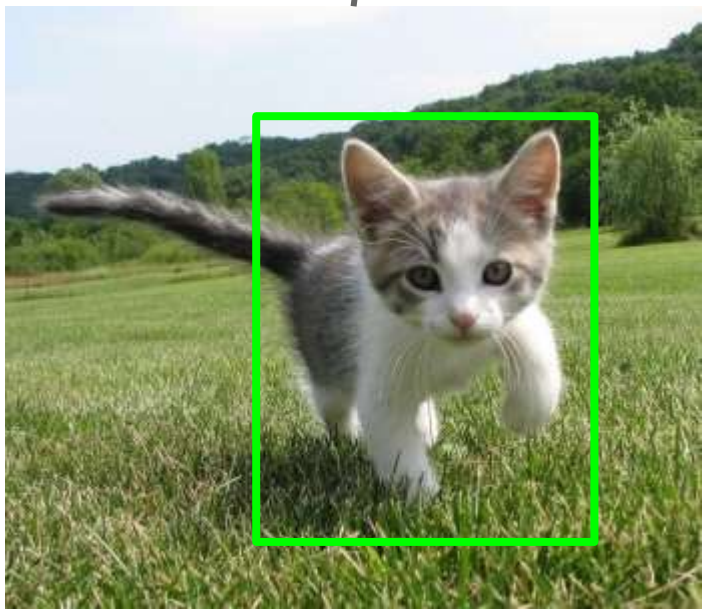
点(x, y) 的特征f_{xy}是其近邻的4个网格特征的线性组合

He et al, “Mask R-CNN”, ICCV 2017

裁剪特征：RoI对齐

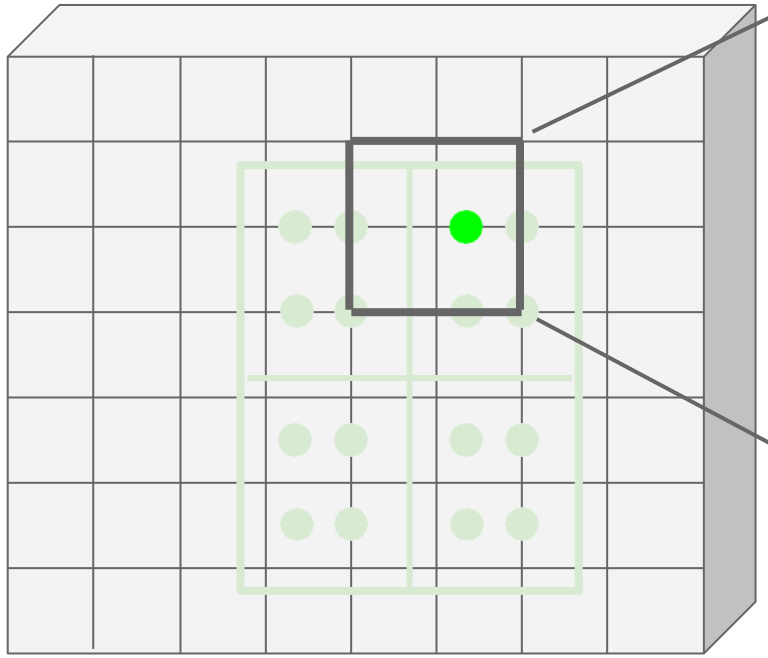
候选区域映射到特征上

没有“对齐”！



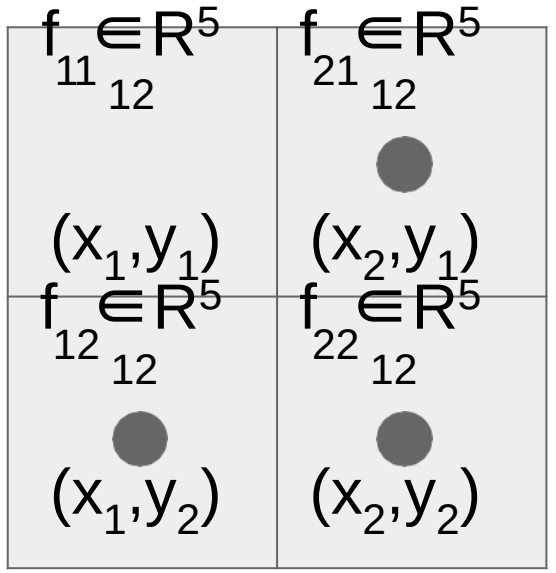
输入图片
(e.g. 3 x 640 x 480)

CNN



图片特征
(e.g. 512 x 20 x 15)

在子区域中用双线性插值法采样



点(x, y) 的特征f_{xy}是其近邻的4个网格特征的线性组合

$$f_{xy} = \sum_{i,j=1}^{2} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_j|)$$

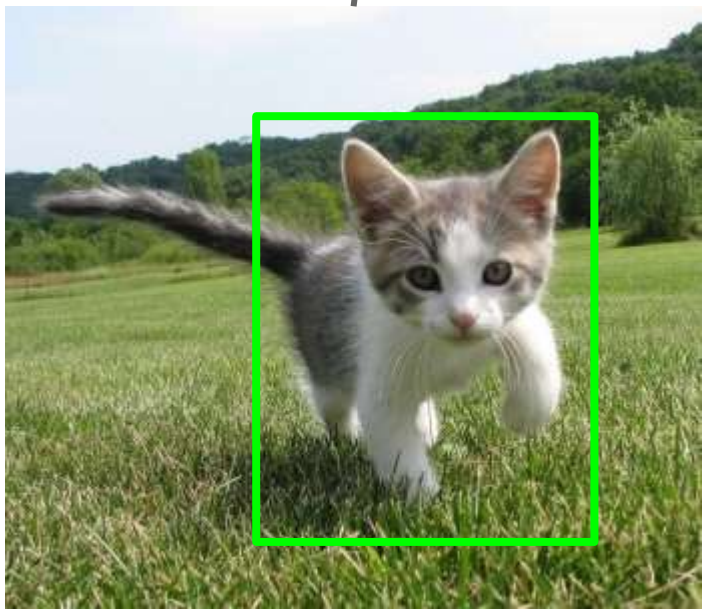
He et al, “Mask R-CNN”, ICCV 2017

裁剪特征：RoI对齐

候选区域映射到特征上

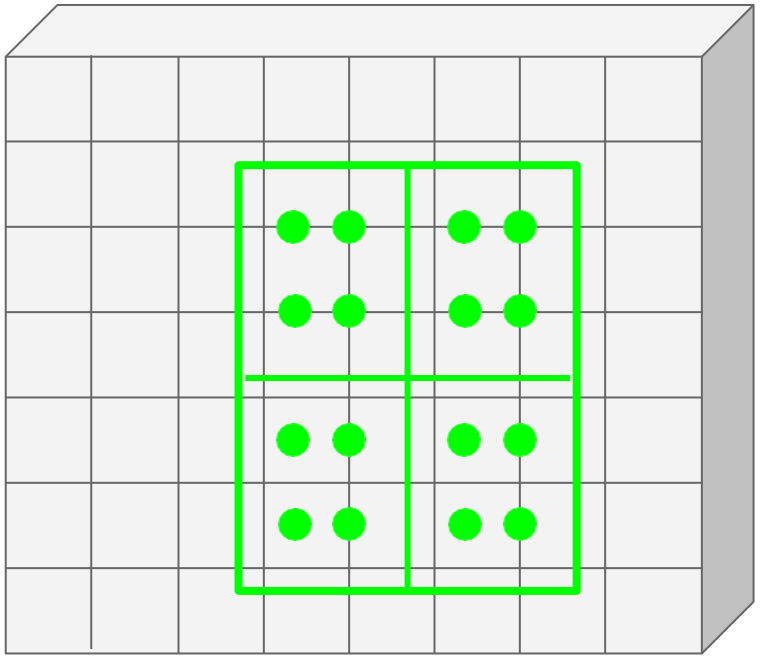
没有“对齐”！

在子区域中用双线性插值法采样



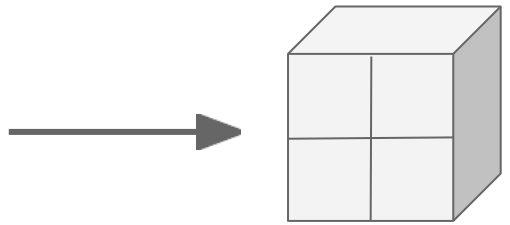
输入图片
(e.g. 3 x 640 x 480)

CNN



图片特征
(e.g. 512 x 20 x 15)

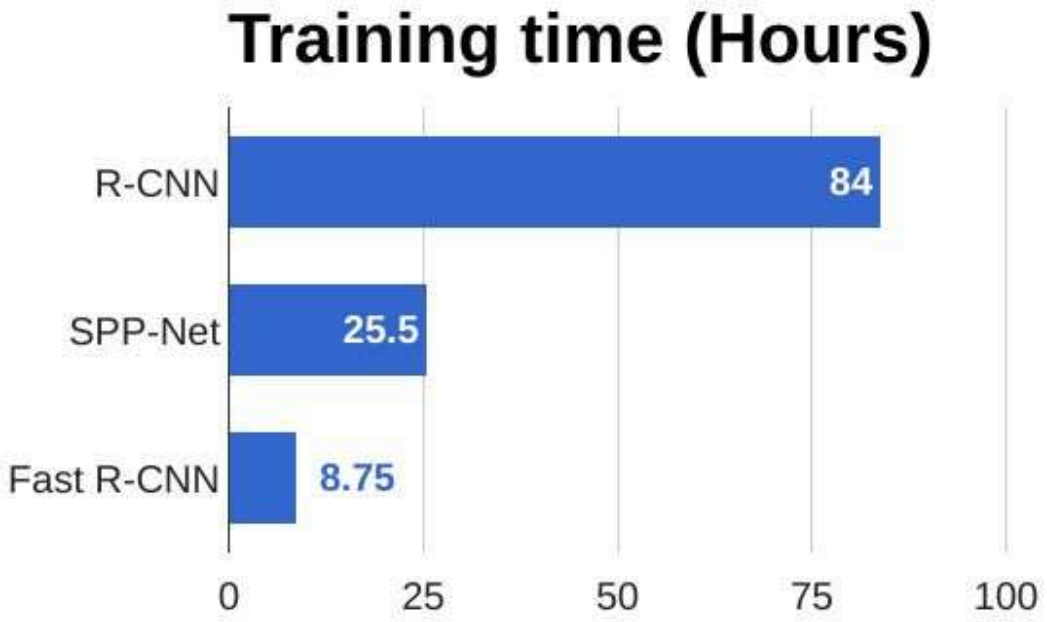
每个子区域内
做最大池化

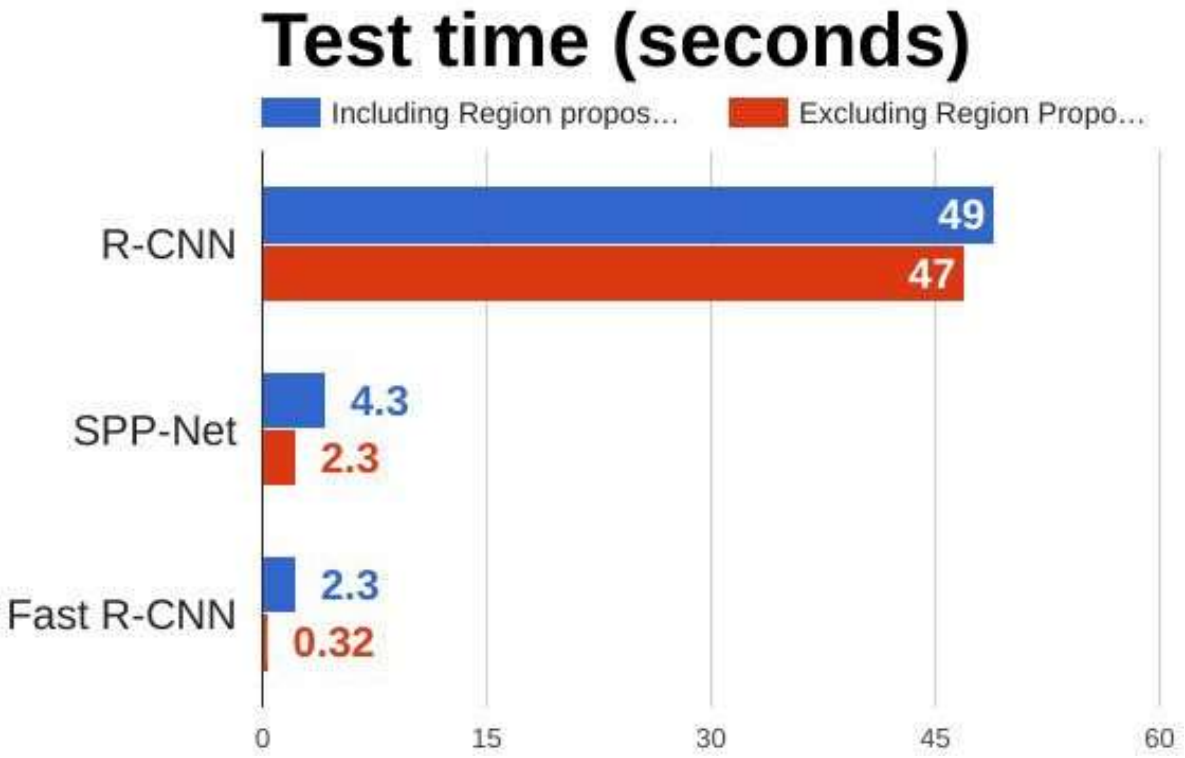


区域特征(这里是512 x 2 x 2;
实际操作中可能是512 x 7 x 7)

He et al, “Mask R-CNN”, ICCV 2017

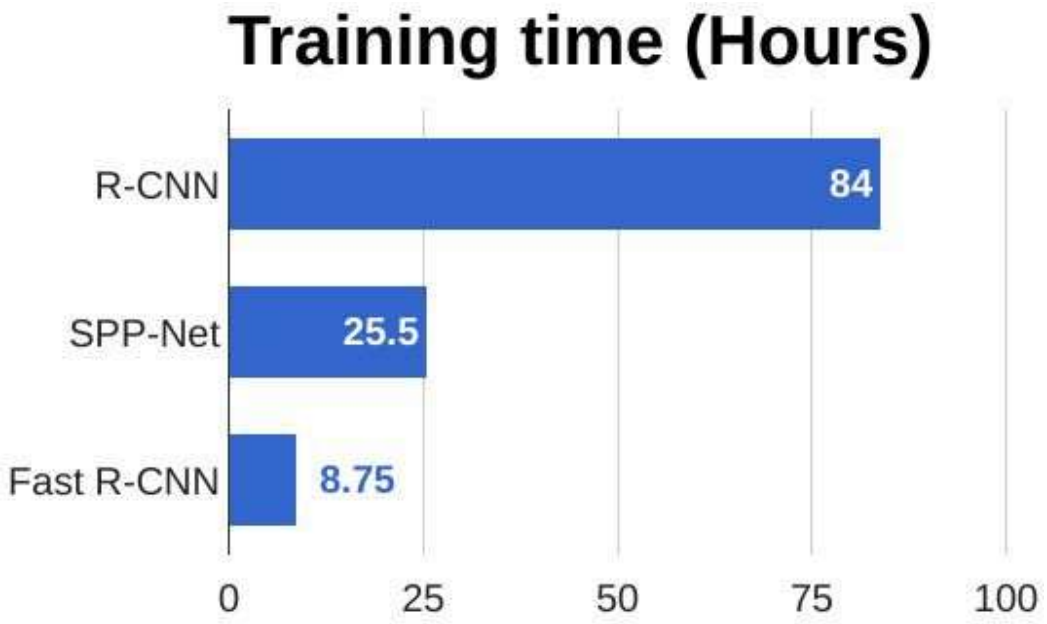
R-CNN vs Fast R-CNN

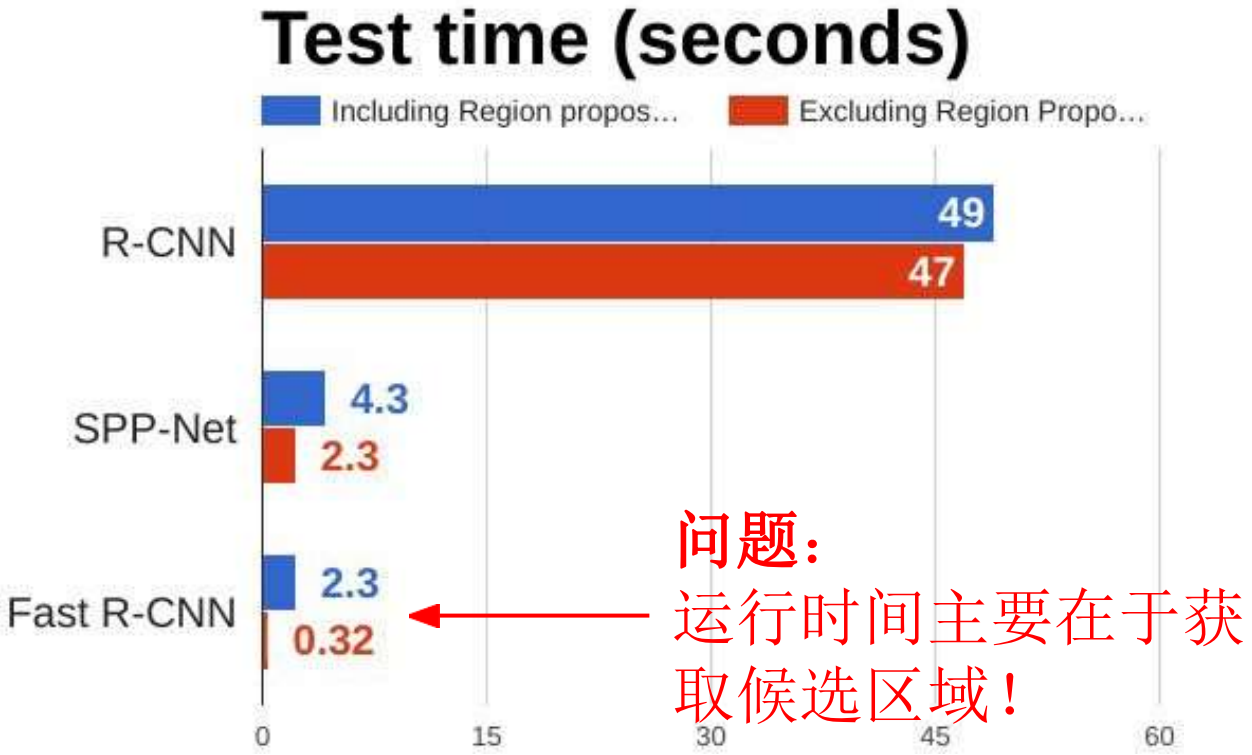




Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014. He et al, “Spatial pyramid pooling in deep convolutional networks for visual recognition”, ECCV 2014
Girshick, “Fast R-CNN”, ICCV 2015

R-CNN vs Fast R-CNN





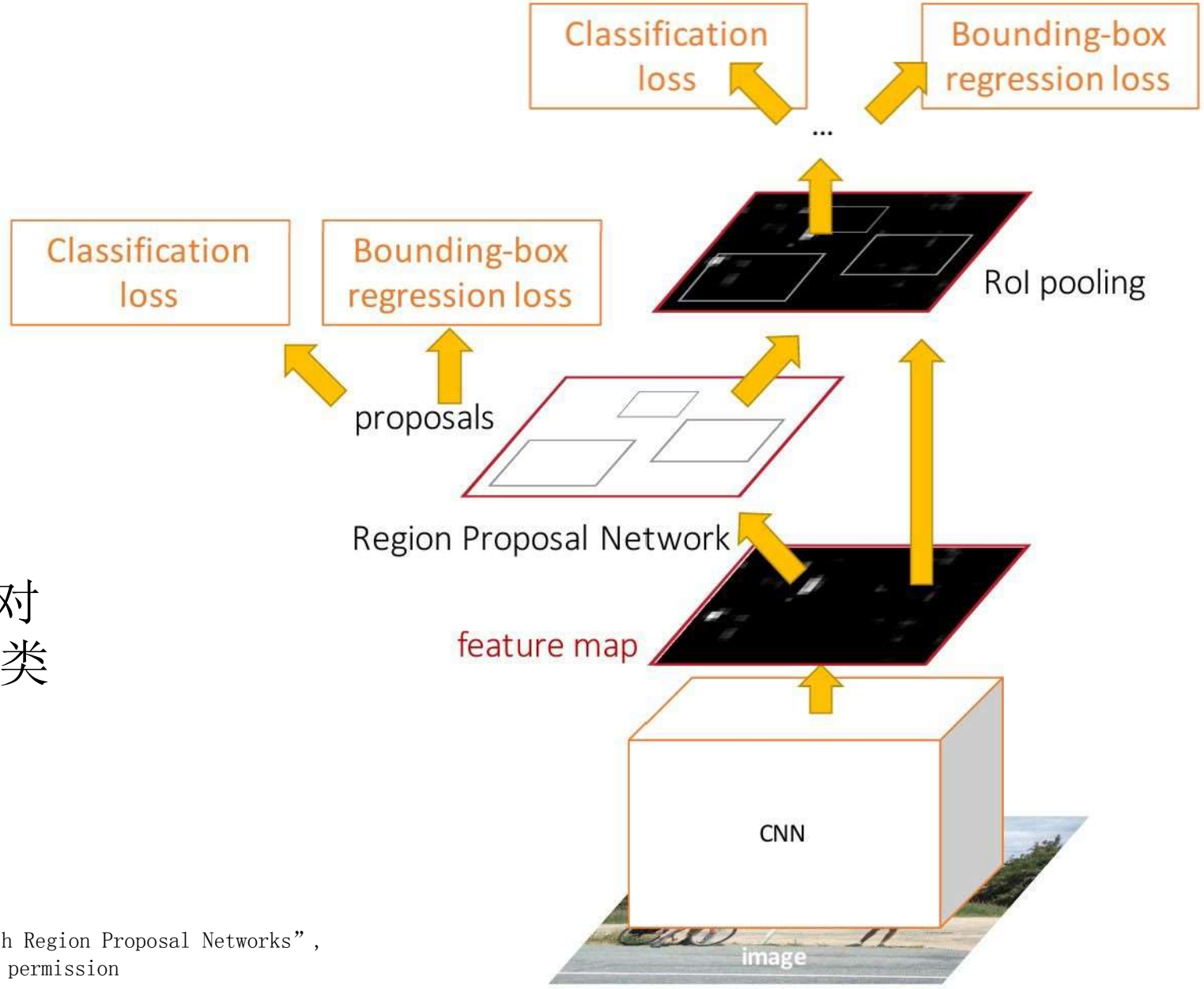
问题：
运行时间主要在于获取候选区域！

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014. He et al, “Spatial pyramid pooling in deep convolutional networks for visual recognition”, ECCV 2014
Girshick, “Fast R-CNN”, ICCV 2015

Faster R-CNN: 让CNN做推荐!

插入区域候选网络(PRN)来从特征预测候选区域

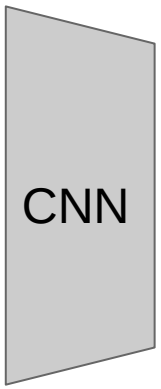
其他地方和Fast R-CNN相同: 对于每个候选区域裁剪特征并分类

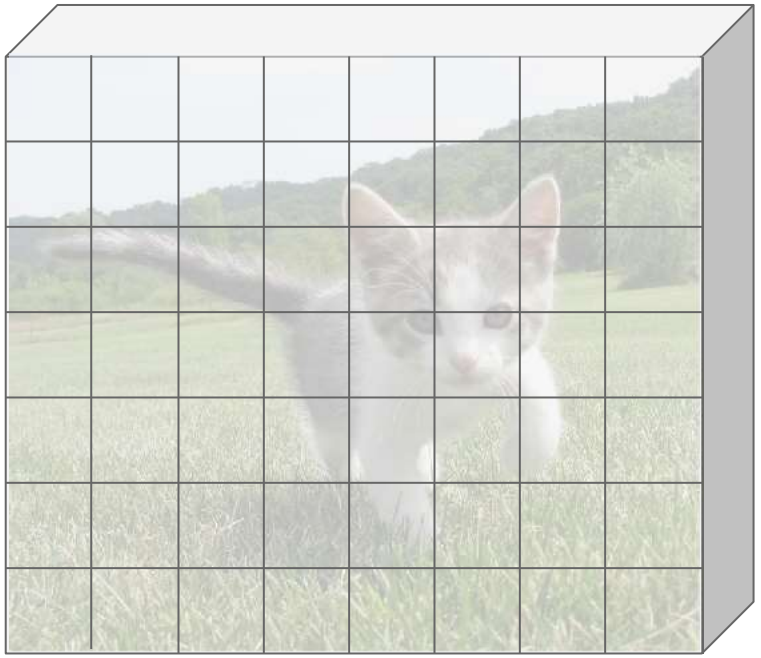


Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015 Figure copyright 2015, Ross Girshick; reproduced with permission

区域候选网络(Region Proposal Network)

输入图片
(e.g. 3 x 640 x 480)



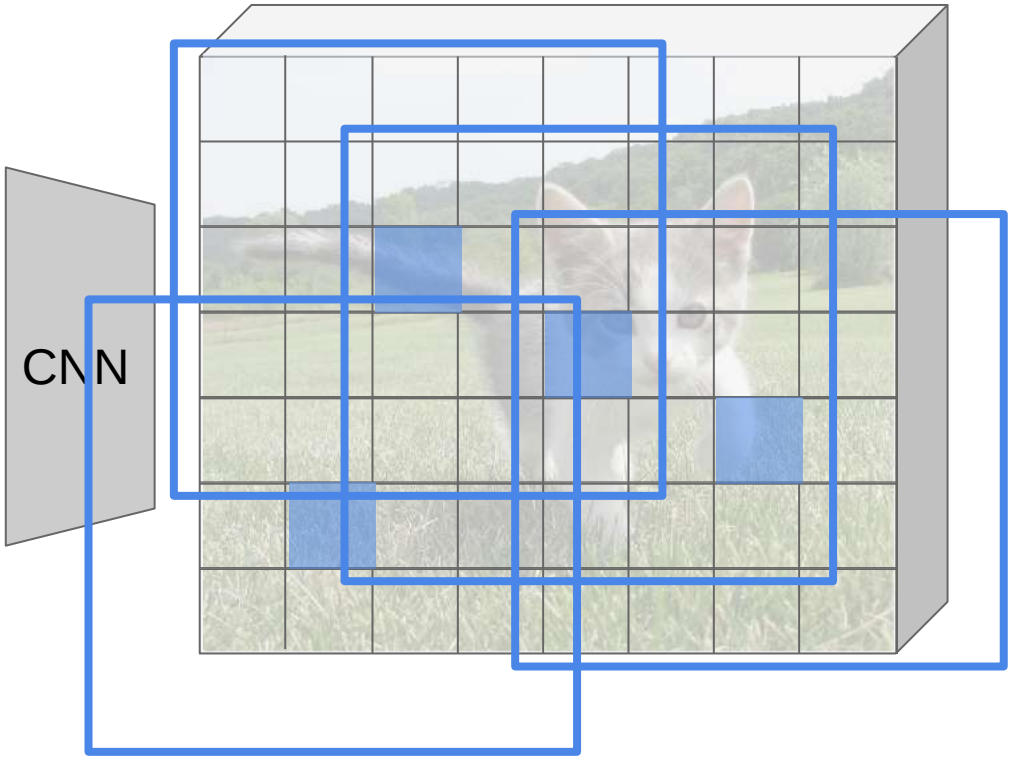


图片特征
(e.g. 512 x 20 x 15)

区域候选网络(RPN)

在特征图中的每个点想象一个固定大小的**锚边框**(anchor box)

输入图片
(e.g. 3 x 640 x 480)

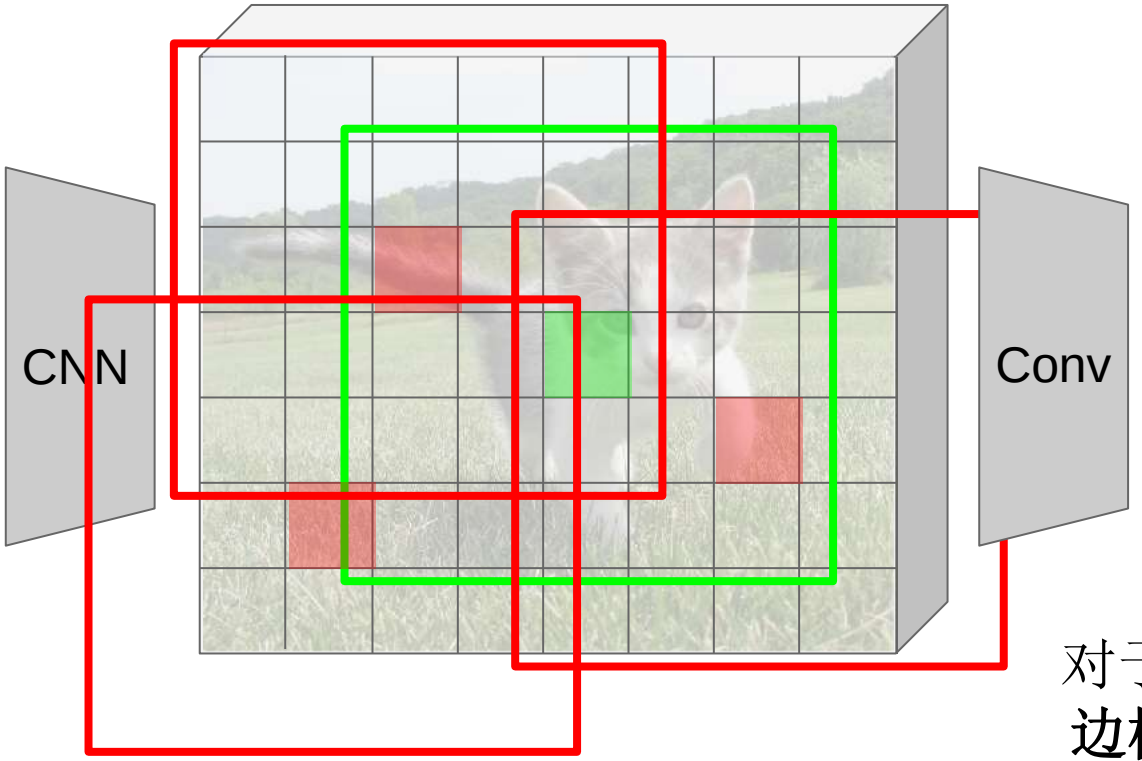


图片特征
(e.g. 512 x 20 x 15)

区域候选网络(RPN)

在特征图中的每个点想象一个固定大小的**锚边框**(anchor box)

输入图片
(e.g. 3 x 640 x 480)



图片特征
(e.g. 512 x 20 x 15)

锚边框是否包含对象?
1 x 20 x 15

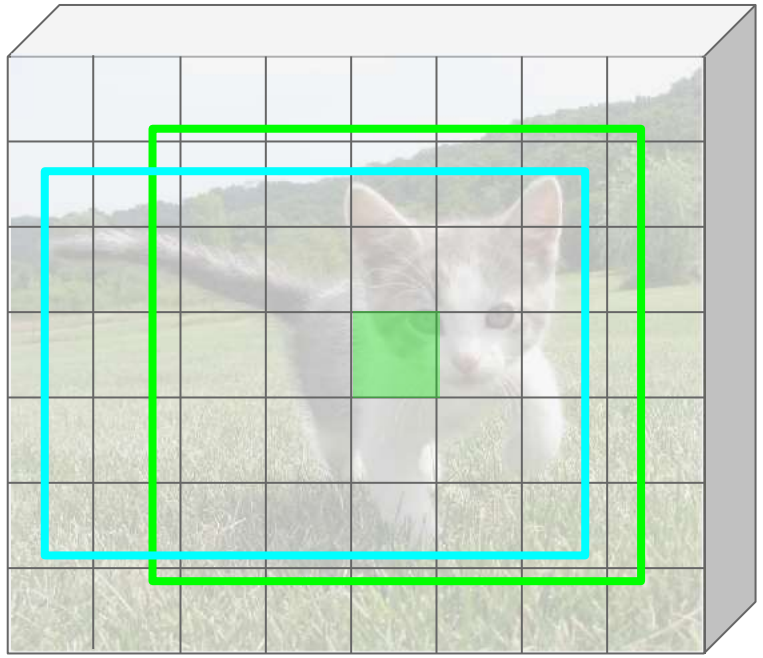
对于每个点，预测相应的**锚边框**是否包含对象(二分类)

区域候选网络(RPN)

在特征图中的每个点想象一个固定大小的**锚边框**(anchor box)

输入图片
(e.g. 3 x 640 x 480)

CNN



图片特征
(e.g. 512 x 20 x 15)

Conv

锚边框是否包含对象?
1 x 20 x 15

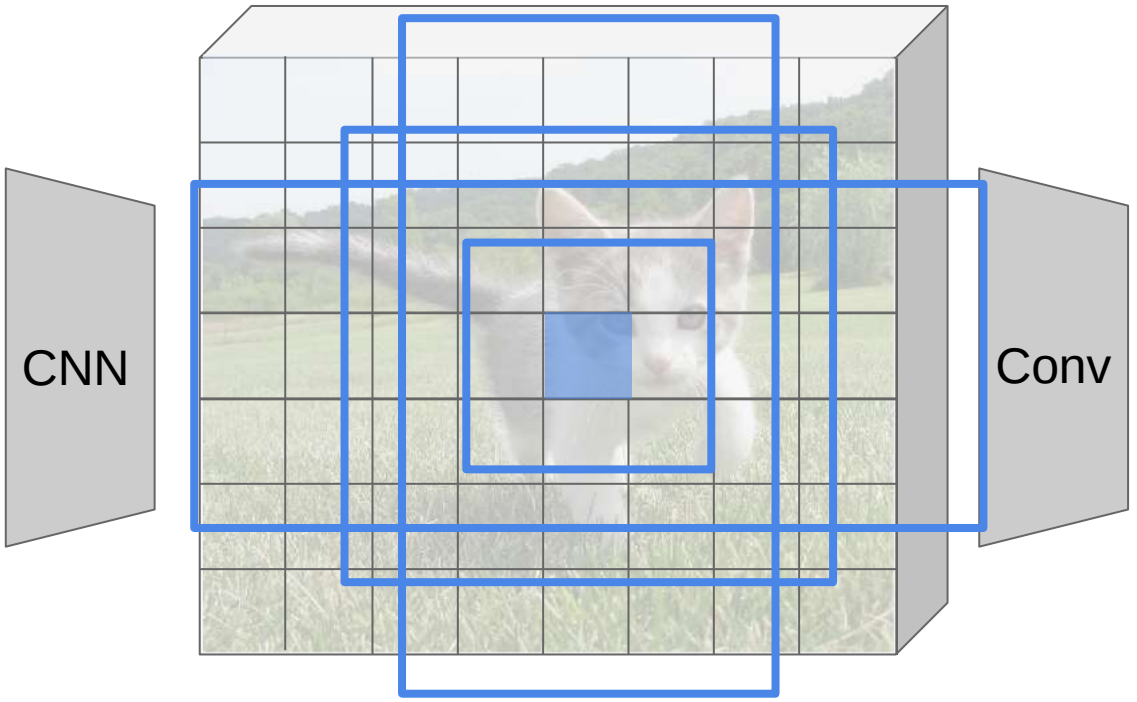
边界框修正
4 x 20 x 15

对于正锚边框，同样计算它和真实边界框的差异从而对四个数值进行回归学习

区域候选网络(RPN)

实际操作中对于每个点作K个不同大小/比例的锚边框

输入图片
(e.g. 3 x 640 x 480)



图片特征
(e.g. 512 x 20 x 15)

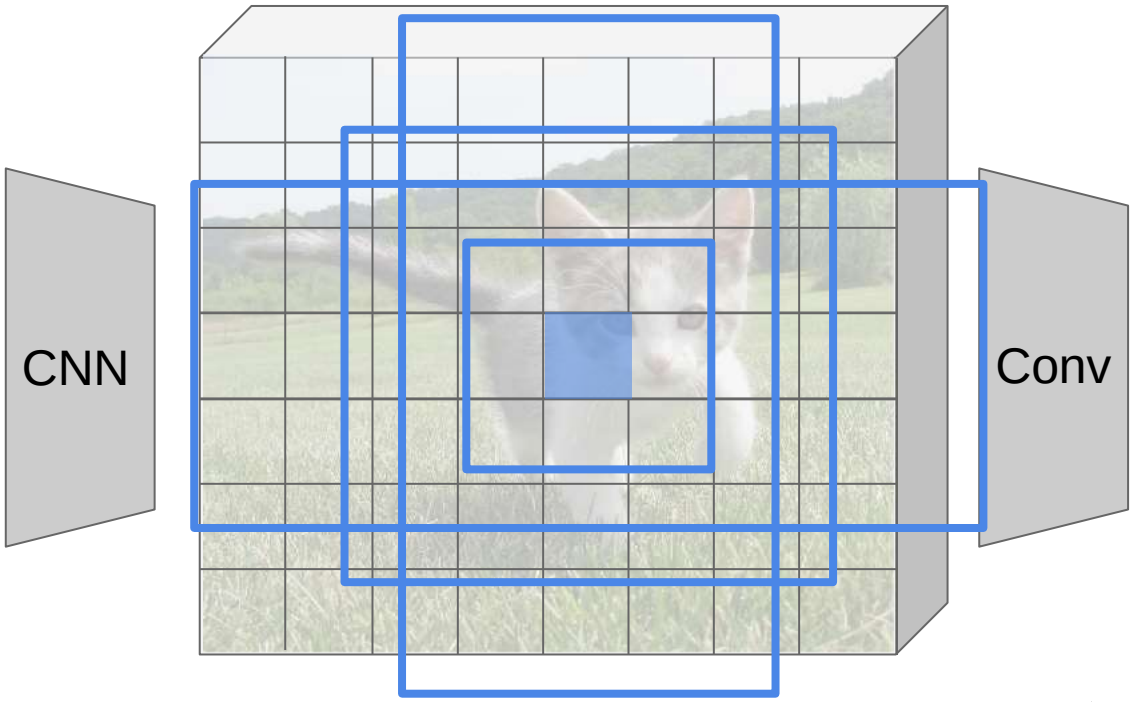
锚边框是否包含对象?
K x 20 x 15

边界框修正
4K x 20 x 15

区域候选网络(RPN)

实际操作中对于每个点作K个不同大小/比例的锚边框

输入图片
(e.g. 3 x 640 x 480)



图片特征
(e.g. 512 x 20 x 15)

锚边框是否包含对象?
K x 20 x 15

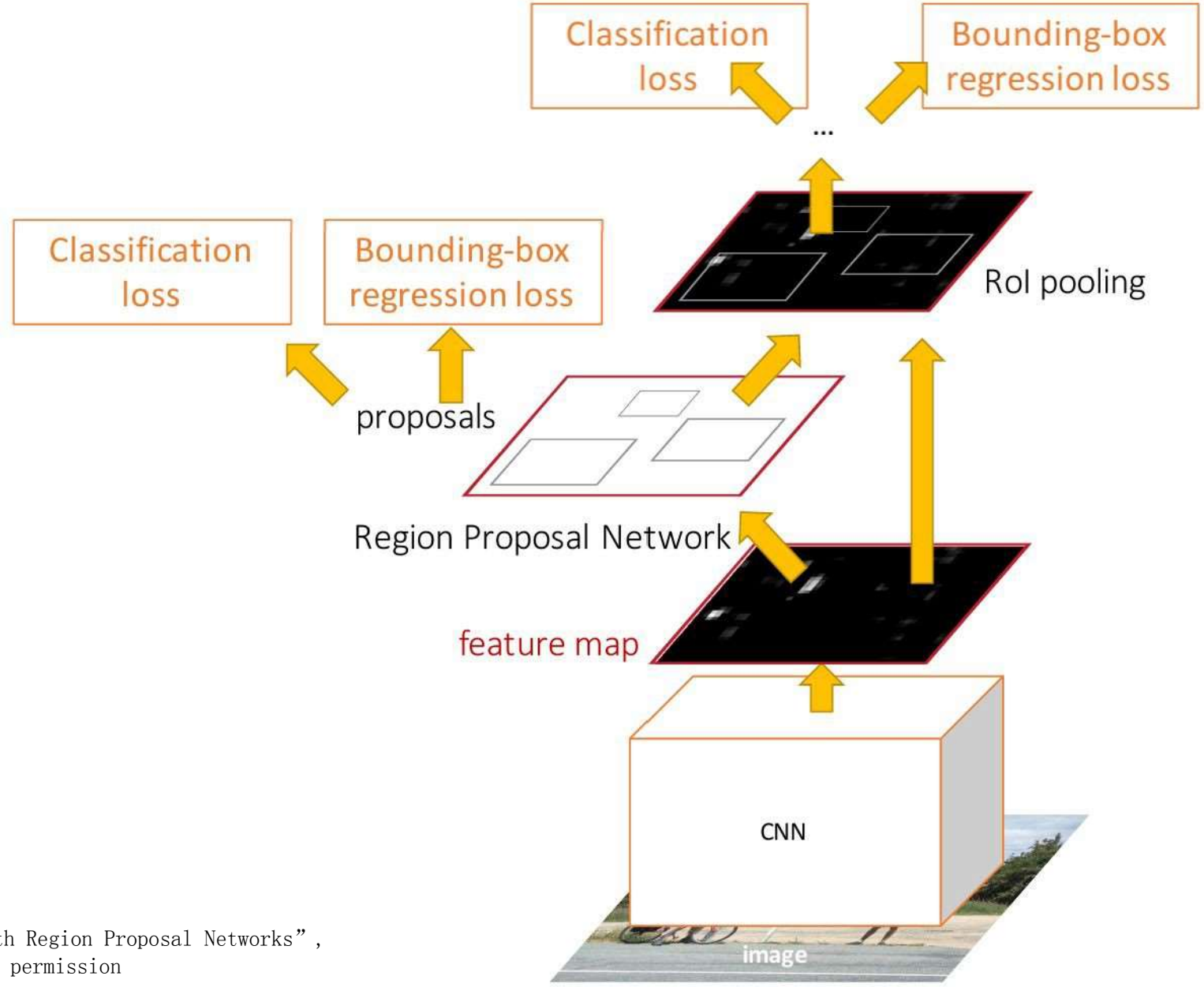
边界框修正
4K x 20 x 15

通过“objectness”分数对这K*20*15个边框排序，取最高的~300个作为最终候选区域

Faster R-CNN:
让CNN做推荐！

用4种损失联合训练:

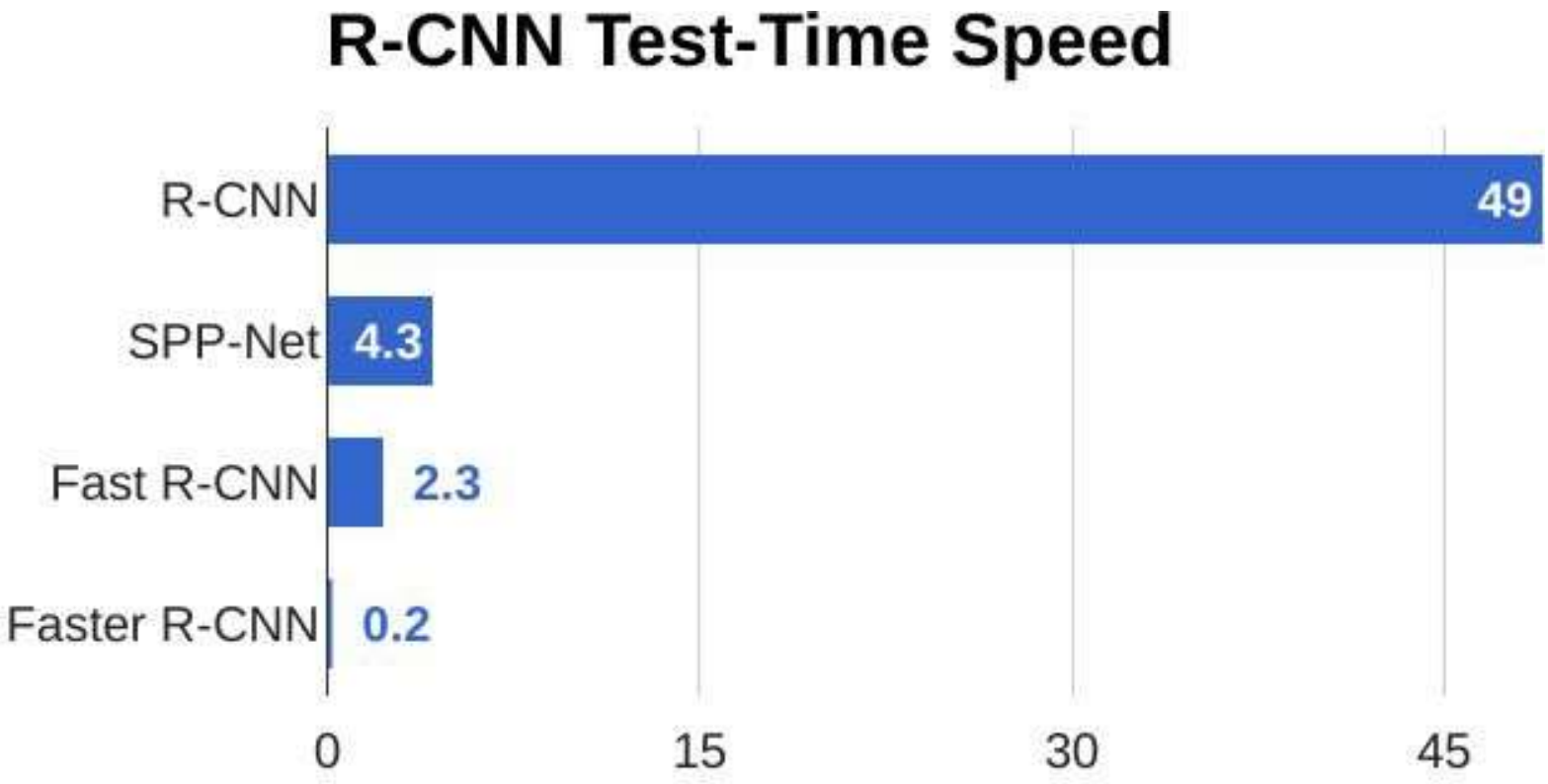
1. RPN分类是否包含对象
2. RPN边界框回归调整
3. 最终分类分数(对象类别)
4. 最终边界框调整



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

Faster R-CNN:

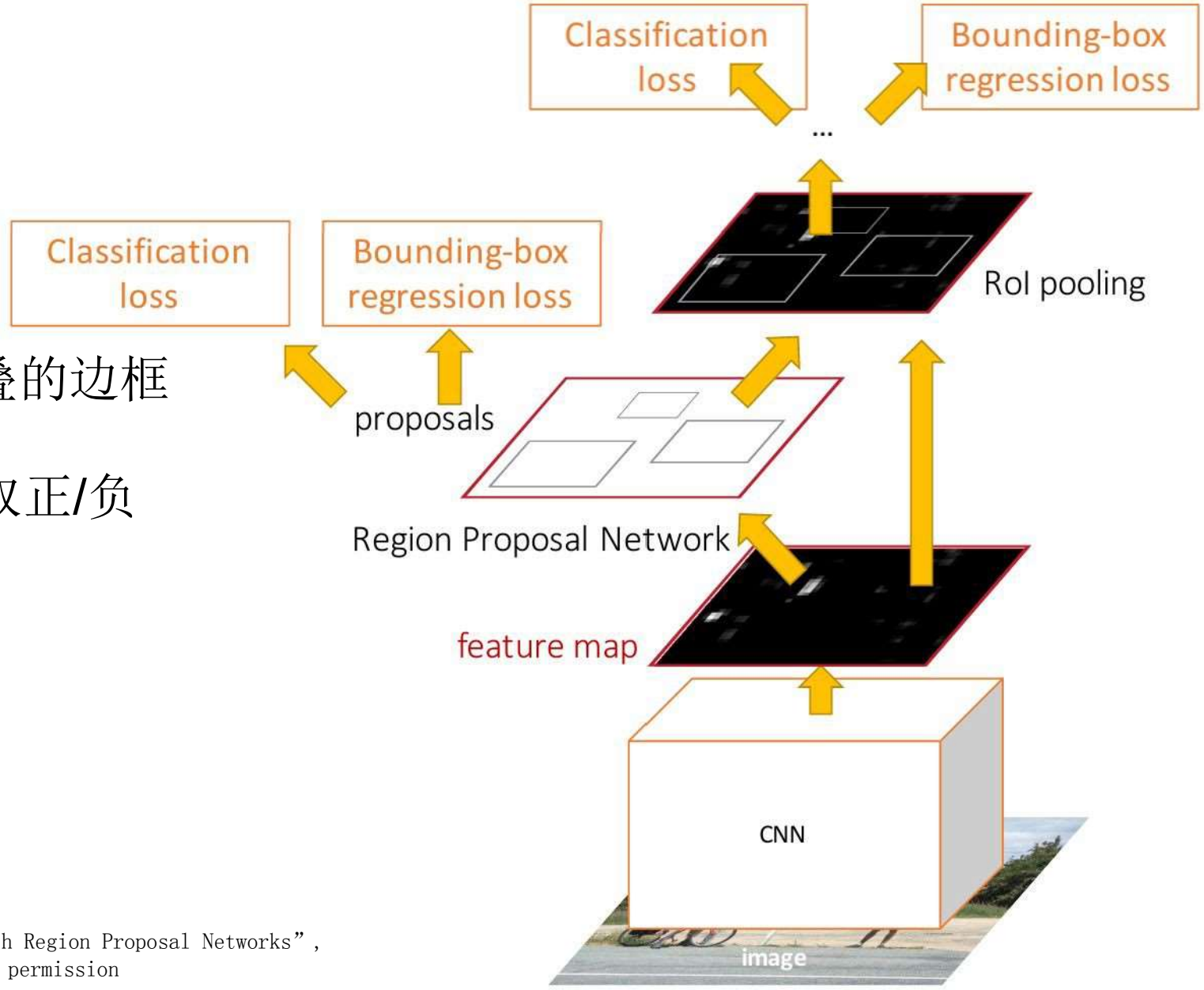
让CNN做推荐！



Faster R-CNN: 让CNN做推荐！

省略了很多细节:

- 使用**非最大值抑制**剔除重叠的边框
- 锚边框是如何决定的？
- 为了训练RPN，我们如何取正/负样本？
- 如何将边界框回归参数化？



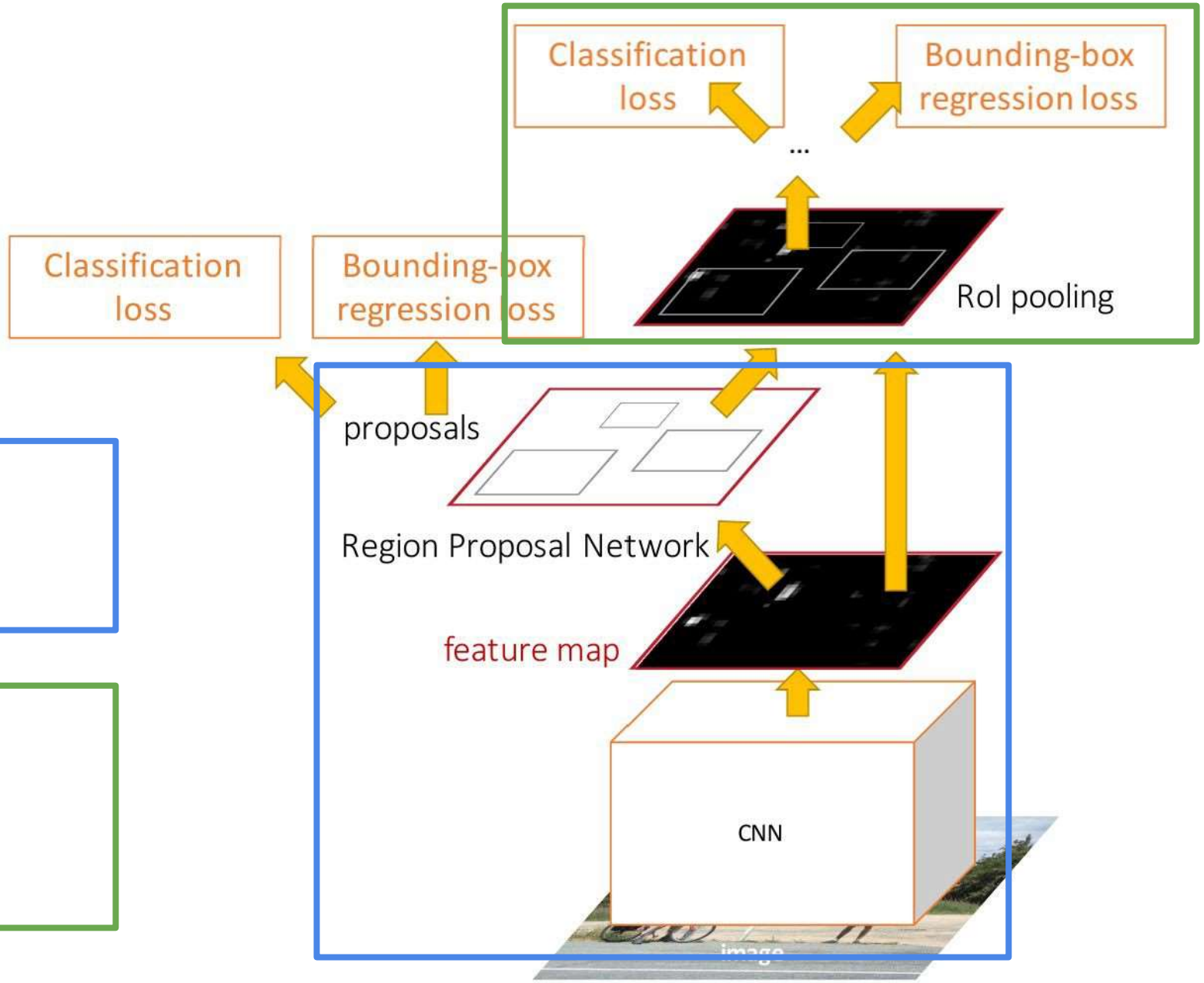
Ren et al, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, NIPS 2015 Figure copyright 2015, Ross Girshick; reproduced with permission

Faster R-CNN: 让CNN做推荐！

Faster R-CNN是一种二阶段目标检测器

一阶段: 每张图片运行一次
- 骨干网络
- RPN

二阶段: 每个区域运行一次
- RoI池化/对齐
- 预测对象类别
- 预测边框偏移



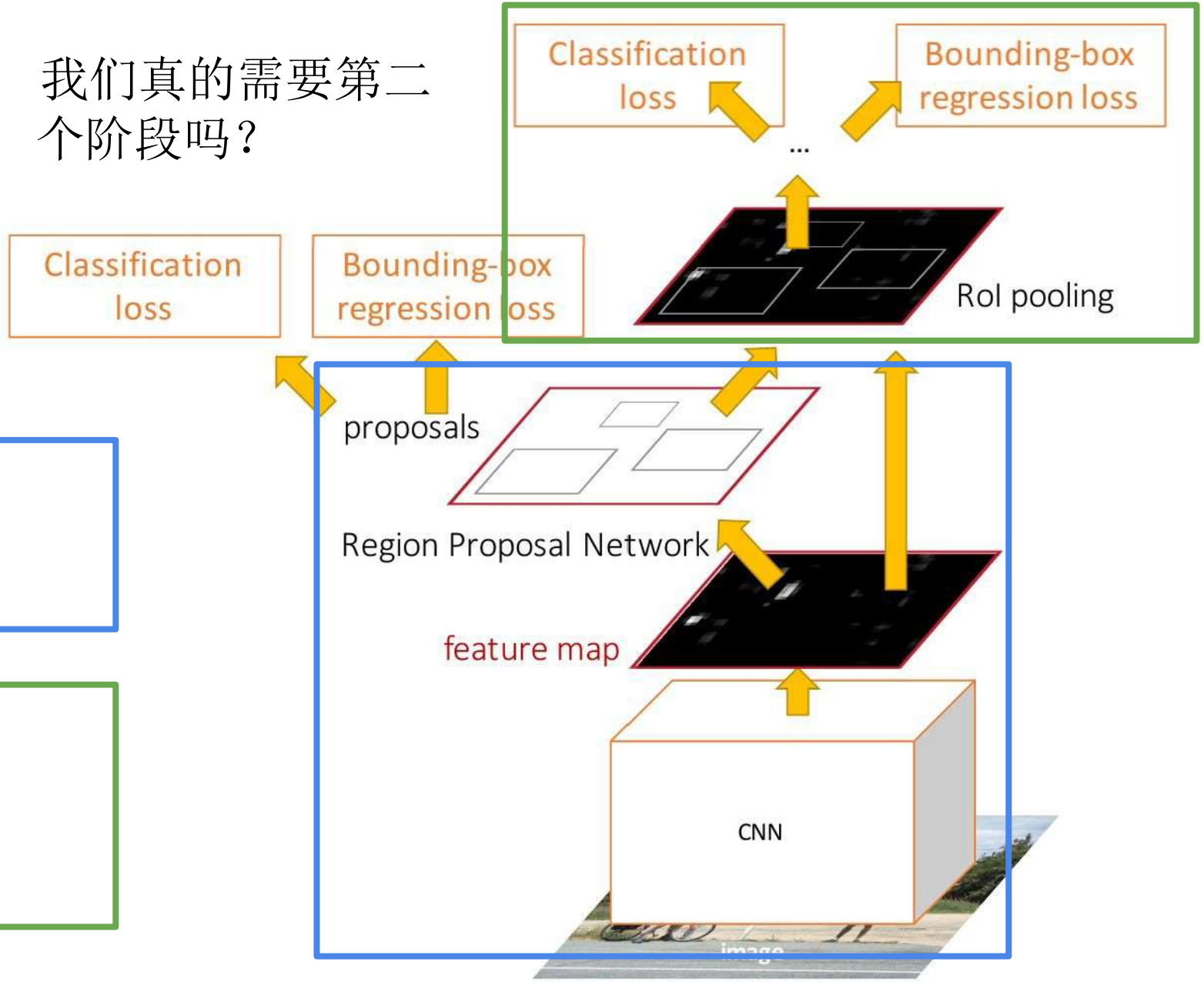
Faster R-CNN:
让CNN做推荐！

我们真的需要第二个阶段吗？

Faster R-CNN是一种**二阶段目标检测器**

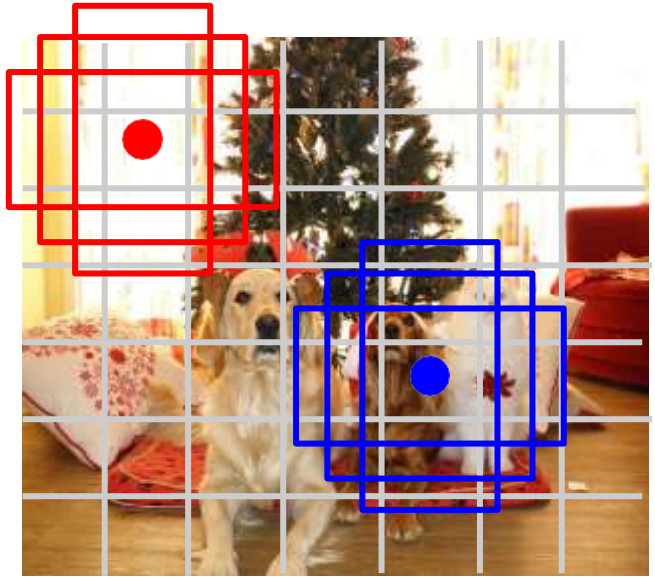
一阶段: 每张图片运行一次
- 骨干网络
- RPN

二阶段: 每个区域运行一次
- RoI池化/对齐
- 预测对象类别
- 预测边框偏移



单阶段目标检测器：YOLO / SSD / RetinaNet

输入图片
3 x H x W



将图片分为7 x 7网格

以每个网格为中心想象一系列**基础边界框**，此处B = 3

在每个网格内：

- 由基础边界框到真实边界框回归训练5个数：
  (dx, dy, dh, dw, 置信度)
- 对C个类别预测分数(将背景也视为一类)
- 与RPN十分相似，但预测了特定的类别！

输出：
7 x 7 x (5 * B + C)

Redmon et al, “You Only Look Once: Unified, Real-Time Object Detection”, CVPR 2016
Liu et al, “SSD: Single-Shot MultiBox Detector”, ECCV 2016 Lin et al, “Focal Loss for Dense Object Detection”, ICCV 2017

目标检测：多种变量 . . .

骨干网络：
VGG16
ResNet-101
Inception V2
Inception V3
Inception
ResNet
MobileNet

主要架构：
二阶段: Faster R-CNN
单阶段: YOLO / SSD
混合: R-FCN

图片尺寸
 # 候选区域
...

额外说明：
Faster R-CNN 比较慢但更精确

SSD快很多但不那么精确

更大/深的骨干网络效果更好

Huang et al, “Speed/accuracy trade-offs for modern convolutional object detectors”, CVPR 2017

R-FCN: Dai et al, “R-FCN: Object Detection via Region-based Fully Convolutional Networks”, NIPS 2016
Inception-V2: Ioffe and Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”, ICML 2015 Inception V3: Szegedy et al, “Rethinking the Inception Architecture for Computer Vision”, arXiv 2016
Inception ResNet: Szegedy et al, “Inception-V4, Inception-ResNet and the Impact of Residual Connections on Learning”, arXiv 2016
MobileNet: Howard et al, “Efficient Convolutional Neural Networks for Mobile Vision Applications”, arXiv 2017

目标检测：多种变量 ...

骨干网络：
VGG16
ResNet-101
Inception V2
Inception V3
Inception
ResNet
MobileNet

主要架构：
二阶段: Faster R-CNN
单阶段: YOLO / SSD
混合: R-FCN

图片尺寸
 # 候选区域
...

额外说明：
Faster R-CNN 比较慢但更精确

SSD快很多但不那么精确

更大/深的骨干网络效果更好

Huang et al, “Speed/accuracy trade-offs for modern convolutional object detectors”, CVPR 2017

Zou et al, “Object Detection in 20 Years: A Survey”, arXiv 2019

R-FCN: Dai et al, “R-FCN: Object Detection via Region-based Fully Convolutional Networks”, NIPS 2016
Inception-V2: Ioffe and Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”, ICML 2015 Inception V3: Szegedy et al, “Rethinking the Inception Architecture for Computer Vision”, arXiv 2016
Inception ResNet: Szegedy et al, “Inception-V4, Inception-ResNet and the Impact of Residual Connections on Learning”, arXiv 2016
MobileNet: Howard et al, “Efficient Convolutional Neural Networks for Mobile Vision Applications”, arXiv 2017