2 图像分类

数据驱动的方法：通过给定数据集进行分类学习，然后对新的图片进行预测得到结果

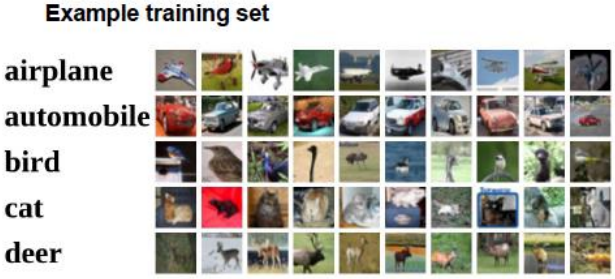
Machine Learning: Data-Driven Approach

1. Collect a dataset of images and labels
2. Use Machine Learning algorithms to train a classifier
3. Evaluate the classifier on new images

```
def train(images, labels):
  # Machine learning!
  return model
```

```
def predict(model, test_images):
  # Use model to predict labels
  return test_labels
```

Example training set



2.1 K 最近邻算法 K Nearest Neighbor Classifier

输入图片，在训练集中找到与该图片最相近的 K 张图片，根据多数投票决定其类型

- **图片距离度量（如何判断为最相近）：**

（1）L1 距离：将两张图片的像素相减取绝对值，然后相加得到距离值

Distance Metric to compare images

L1 distance: $d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$

test image			
56	32	10	18
90	23	128	133
24	26	178	200
2	0	255	220

\-

training image			
10	20	24	17
8	10	89	100
12	16	178	170
4	32	233	112

=

pixel-wise absolute value differences			
46	12	14	1
82	13	39	33
12	10	0	30
2	32	22	108

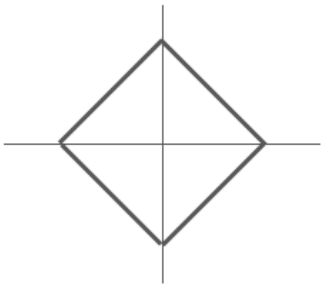
add → 456

（2）L2 距离：将像素相减求平方，然后相加

K-Nearest Neighbors: Distance Metric

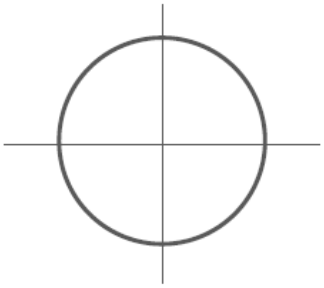
L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$



L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$



- **代码实现：**

```
import numpy as np

class NearestNeighbor:
  def __init__(self):
    pass

  def train(self, X, y):
    """ X is N x D where each row is an example. Y is 1-dimension of size N """
    # the nearest neighbor classifier simply remembers all the training data
    self.Xtr = X
    self.ytr = y

  def predict(self, X):
    """ X is N x D where each row is an example we wish to predict label for """
    num_test = X.shape[0]
    # lets make sure that the output type matches the input type
    Ypred = np.zeros(num_test, dtype = self.ytr.dtype)

    # loop over all test rows
    for i in xrange(num_test):
      # find the nearest training image to the i'th test image
      # using the L1 distance (sum of absolute value differences)
      distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
      min_index = np.argmin(distances) # get the index with smallest distance
      Ypred[i] = self.ytr[min_index] # predict the label of the nearest example

    return Ypred
```

Nearest Neighbor classifier

Q: With N examples, how fast are training and prediction?

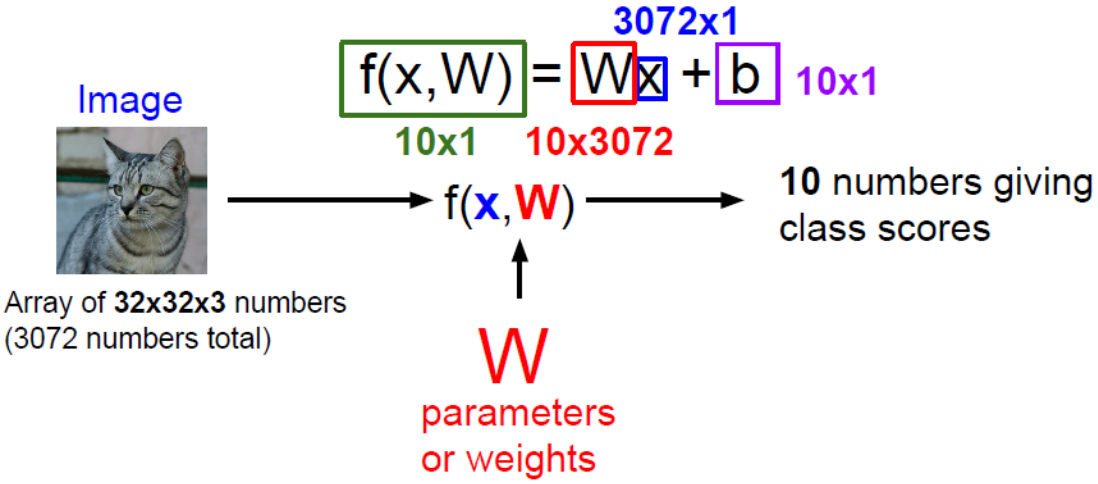
Ans: Train O(1), predict O(N)

This is bad: we want classifiers that are **fast** at prediction; **slow** for training is ok

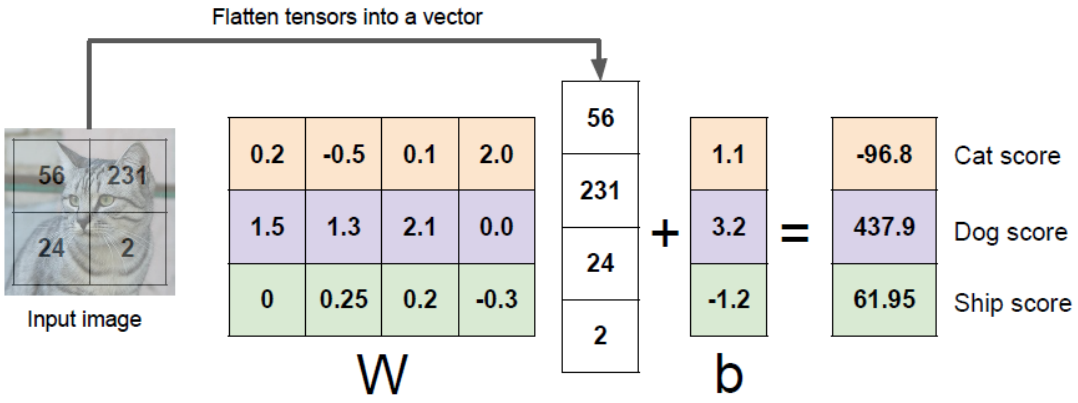
2.2 线性分类算法 Linear Classifier

将图片像素展开为向量，放入线性分类器训练得到参数 W、b；输入新的图片，通过线性分类器得到每一个种类的得分，然后对图片进行分类

Parametric Approach: Linear Classifier

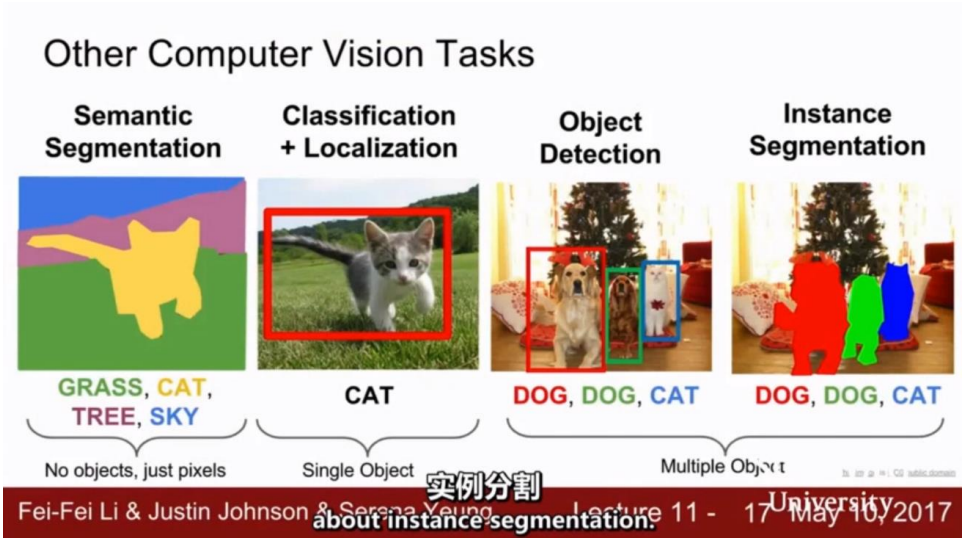


Example with an image with 4 pixels, and 3 classes (cat/dog/ship)



11 图像分割与识别

主要类型：语义分割、图像分类与定位、目标检测、实例分割



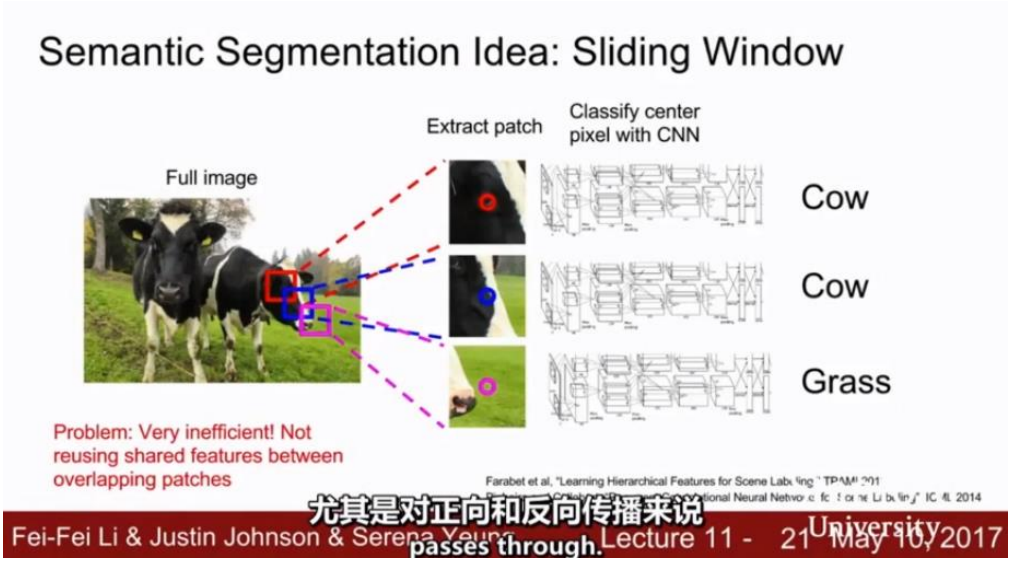
11.1 语义分割 Semantic Segmentation

目的：得到每个像素点的类型

（1）方法一：滑动窗口法

判断每个窗口的类型（Cat or Tree or …），作为这个窗口中心像素点的类型

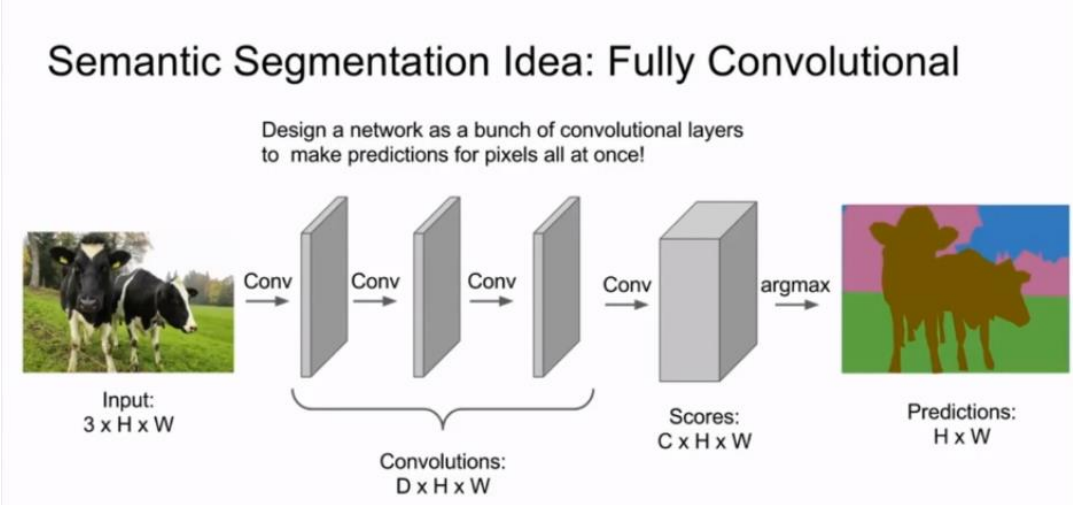
- 缺点：每个像素点需要一个窗口进行判断，计算量过大



（2）方法二：全卷积法 Fully Convolutional

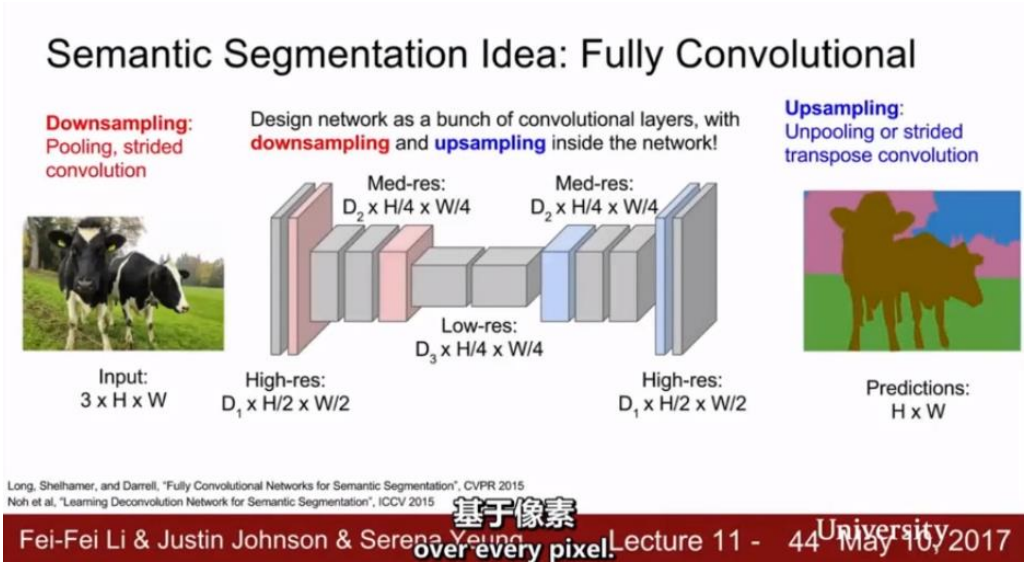
将图片放入全卷积层，通过 padding 保持特征图大小不变，最后得到的特征图即为图片每个像素点的特征向量（长度为种类数量，值为各类的得分），取最大值作为图片中像素点的类型。

- 缺点：每层进行全卷积计算量过大，消耗内存



（3）方法三：全卷积法改进

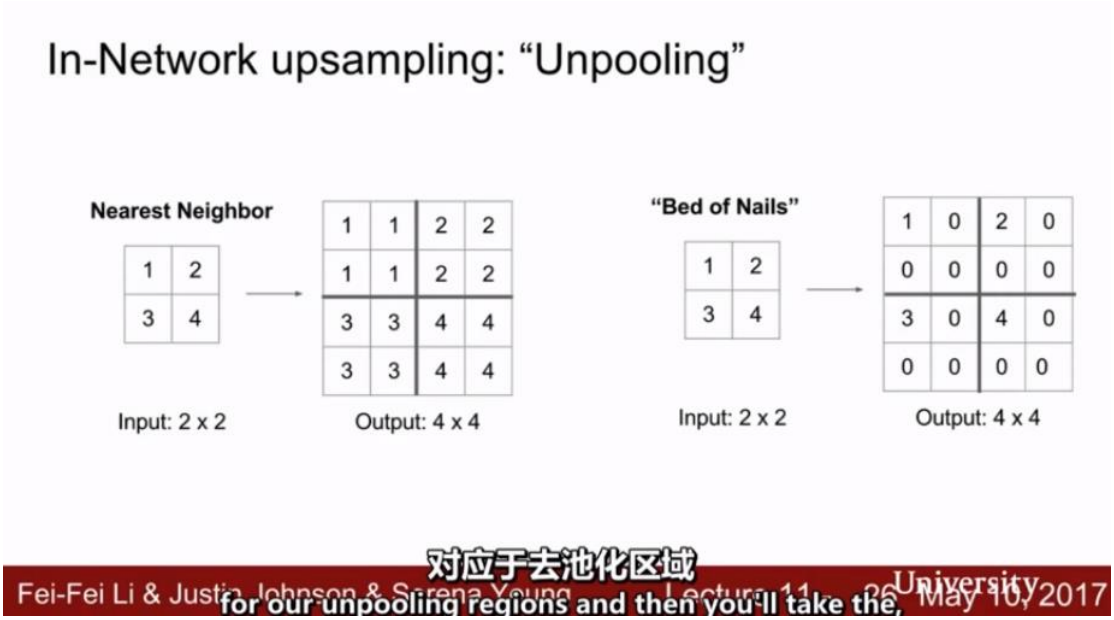
先对图片进行下采样，然后将得到的特征图进行**上采样**，变会原图像的尺寸大小，得到每个像素的类型



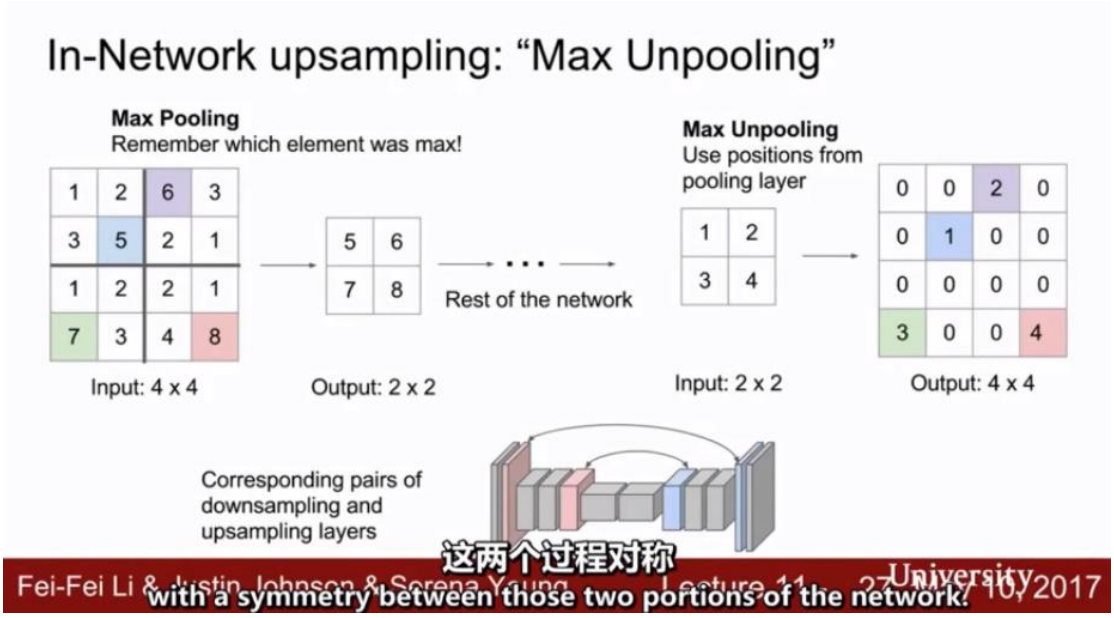
- **上采样：**

1）池化层上采样：去池化 Unpooling

常用方法：Nearest Neighbor、Bed of Nail（Max Unpooling）

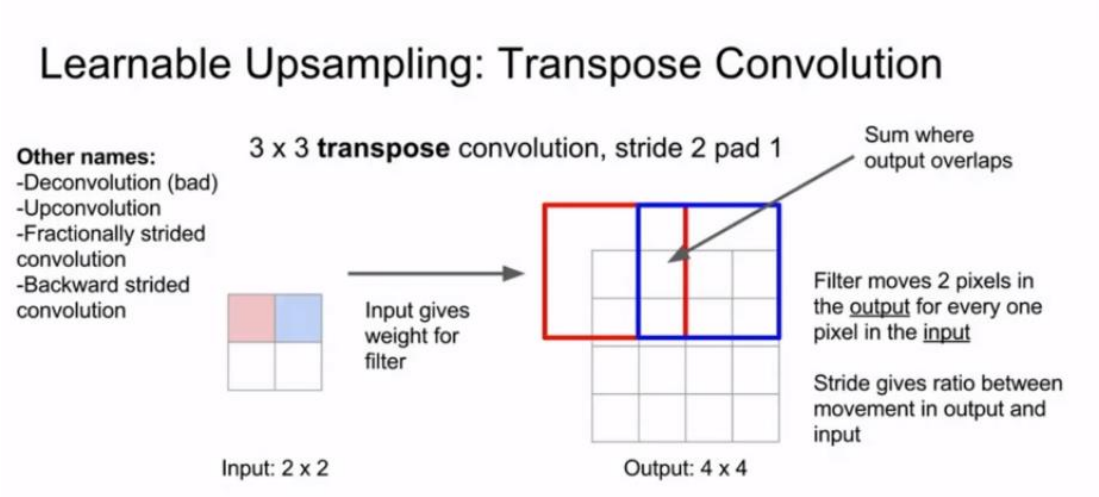


可以与下采样时的 Pooling 对应起来，即 Pooling 时的最大位置作为 Unpooling 时取值的位置



2）卷积层下采样：转置卷积 Transpose Convolution

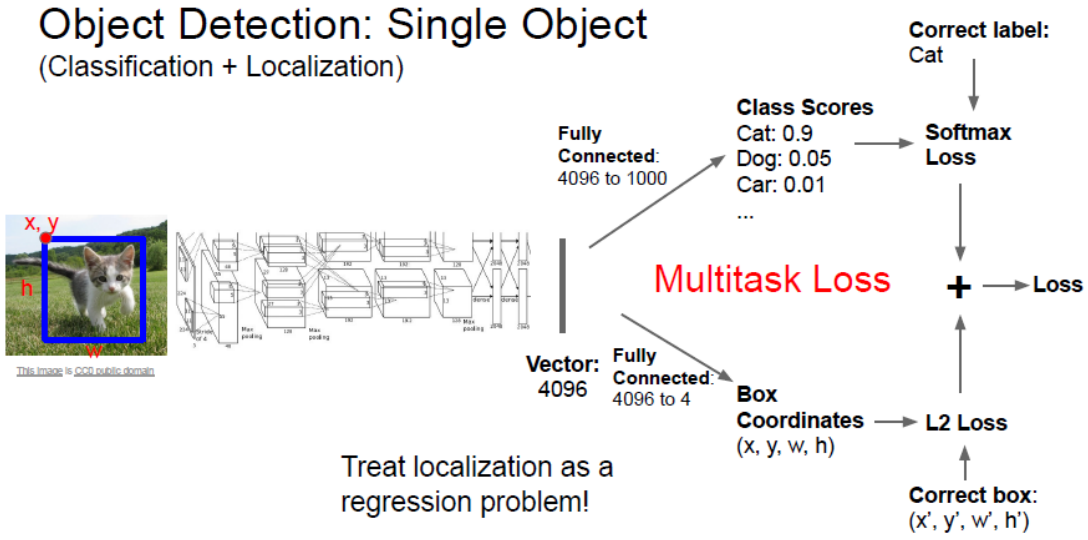
反向卷积操作，将特征图每个位置（标量）与对应卷积的 filter（权值）相乘，放入原图相应位置，位置相同区域进行数值累加（类似于卷积层的反向传播，详见笔记）。



11.2 目标分类与定位 Object Classification & Localization

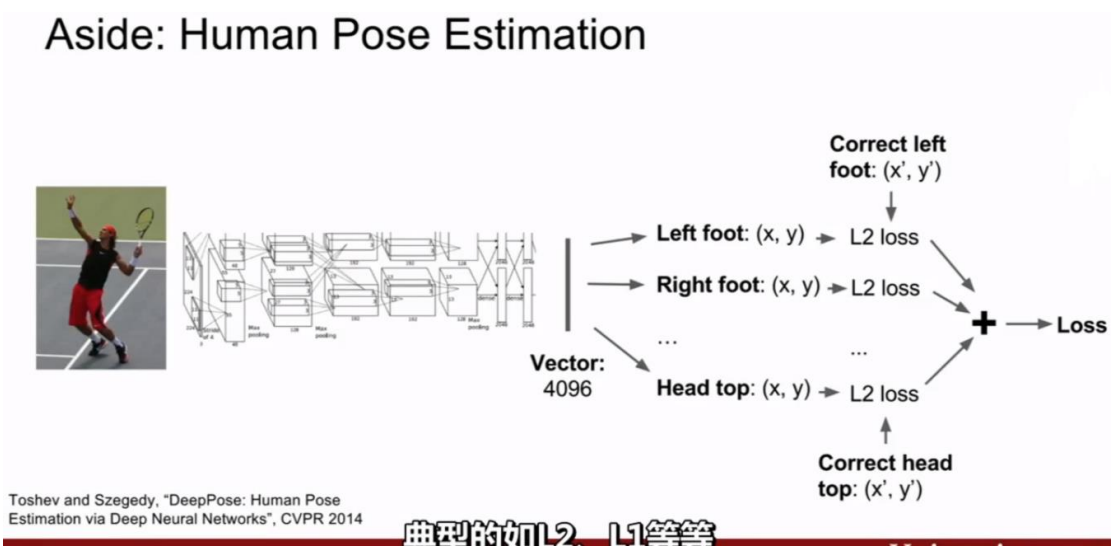
（1）目标分类与定位：单目标检测（图片中只有一个目标）

将图片放入 AlexNet 等卷积神经网络中得到特征向量，将该向量分别放入两个全连接层，其中一个输出每个类型的得分（softmax），另一个输出边界框的坐标（回归问题，L2 Loss）



（2）人体姿态估计 Human Pose Estimation——延伸

与上述目标定位相类似，本质是检测图像中的关键点。预测得到关键点的坐标值（回归问题，L2 Loss）



11.3 目标检测 Object Detection

目标数量不确定

（1）滑动窗口法

选择一定大小、形状的窗口与合适的步长在原图中进行滑动，将每个窗口得到的区域放入 CNN 模型进行目标分类与定位。

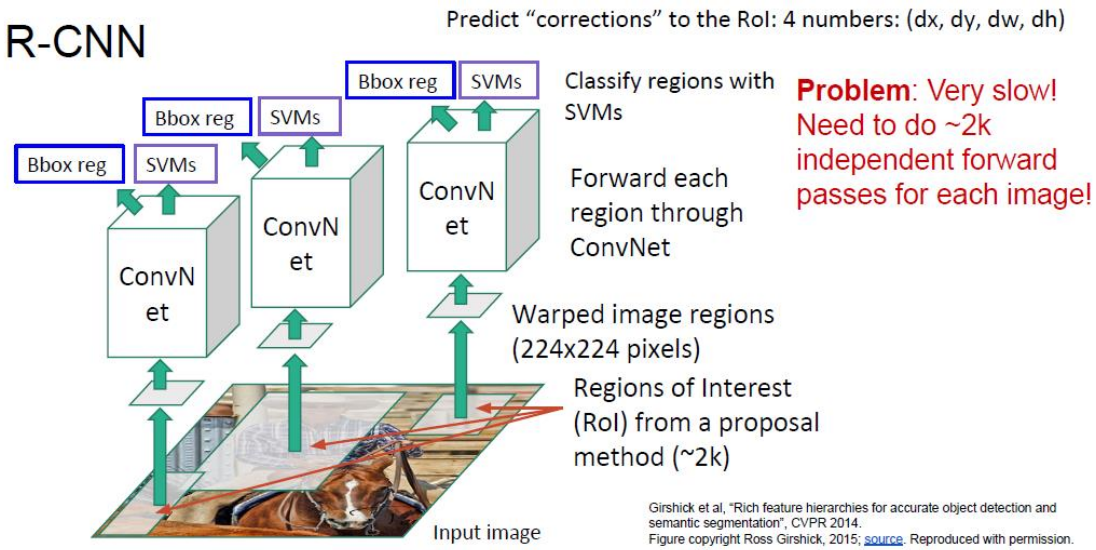
- 缺点：滑动窗口数量众多，计算量过大

（2）基于两阶段的方法（基于候选区域）

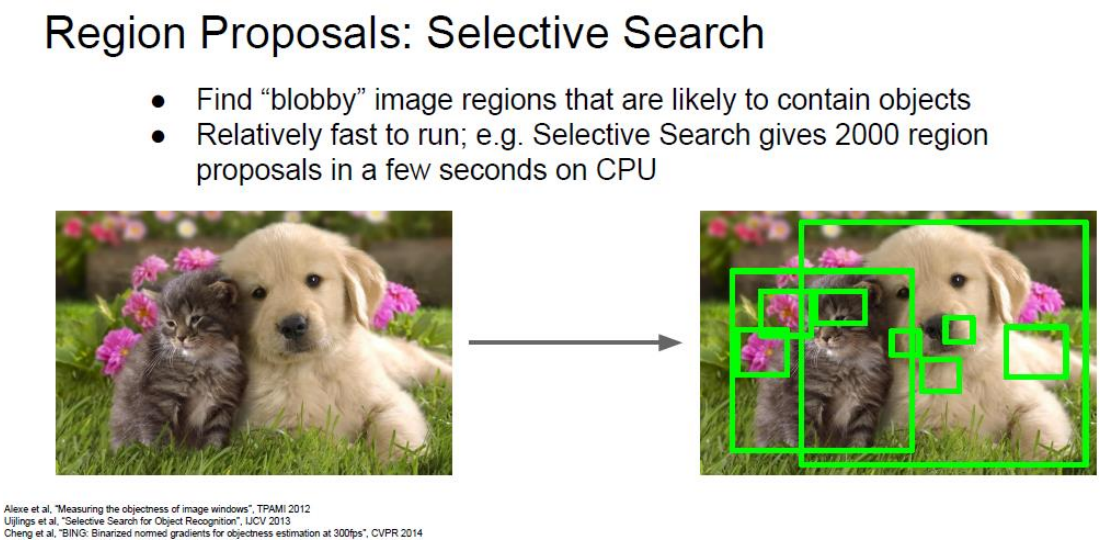
1）R-CNN

先选择 2000 个可能存在目标的 Region Proposals（候选区域），将候选区域的尺寸调整

至相同，然后对每个候选区域进行目标分类（如 SVM）与定位，得到目标类型和边界框。



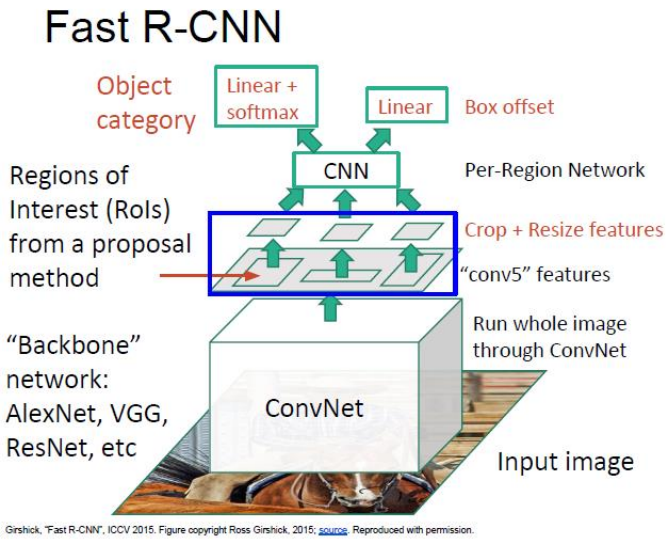
- **候选区域 Region Proposals 选择方法：**采用传统图像处理的方法，寻找图像边界、绘制闭合的边界框

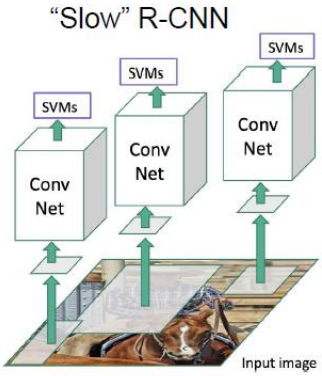


2）Fast R-CNN

先将图片放入 ConvNet 中得到图片的 feature map（特征映射），然后在 feature map 上提取 Region Proposals 对应的卷积块（crops）作为 ROIs，将这些卷积块通过 pooling 统一尺寸后放入 CNN 或全连接层进行目标分类与定位。

- 特点：不在图片上直接提取 Region Proposals 的像素，而是在 feature map 提取属于 Region Proposals 的卷积块，在不同特征映射中共享了训练结果，减少计算量
- 问题：Region Proposals 的提取仍占了大量时间

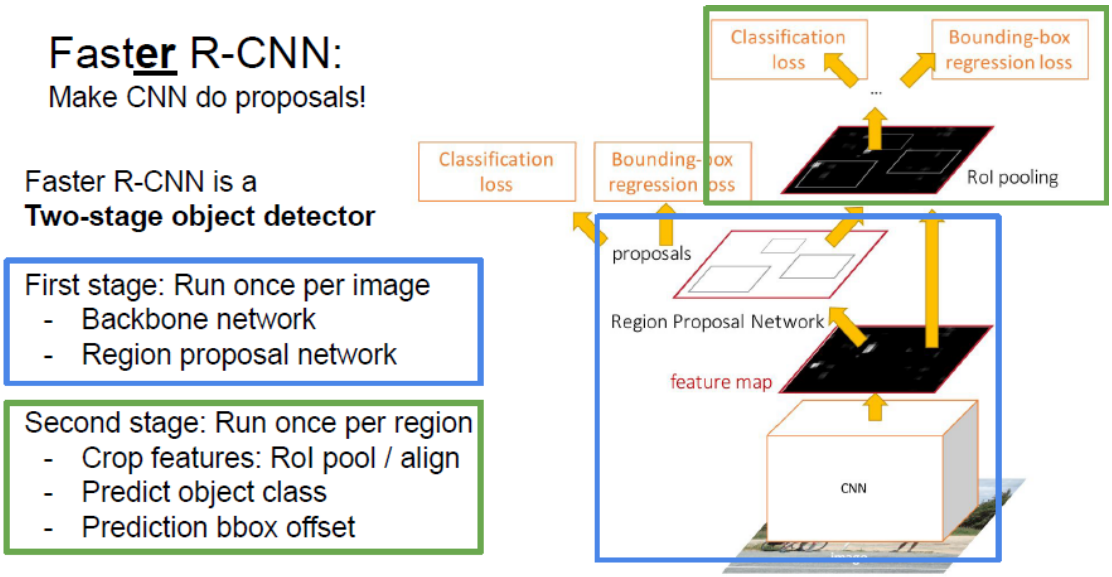




- 卷积块提取方法：ROI Pooling

3）Faster R-CNN

通过 backbone Network（如 VGG16）获取输入图像的 feature map，然后利用 RPN 计算锚框（anchor box）的置信度和调整后的坐标，获取 Region Proposals 及其在 feature map 上对应的卷积块作为 ROIs，将这些卷积块通过 ROIpooling 统一尺寸后放入网络进行目标分类与定位



Region Proposal Network（RPN）：候选区域提取网络

对 feature map 进行滑动窗口处理（即图中的 3×3 卷积处理），每一个窗口都得到一个 256 维特征向量，然后分别放入两个全连接层（即图中的 1×1 卷积），得到 2k 个分数（是否为候选区域）和 4k 个坐标（候选区域边界框的坐标），其中 k 为规定的 anchor box 个数（图中为 9 个）。

每个滑动窗口对应着 k 个 anchor boxes，其中 anchor box 的中心坐标为滑动窗口中心，宽和高为预先定义的 k 种尺寸。训练时计算每个 anchor box 与 ground truth（实际物体框）之间的 IoU，将 IoU 高于 0.7 的 anchor box 标记为正样本，低于 0.3 的标记为负样本（即背景），其余的不用于训练

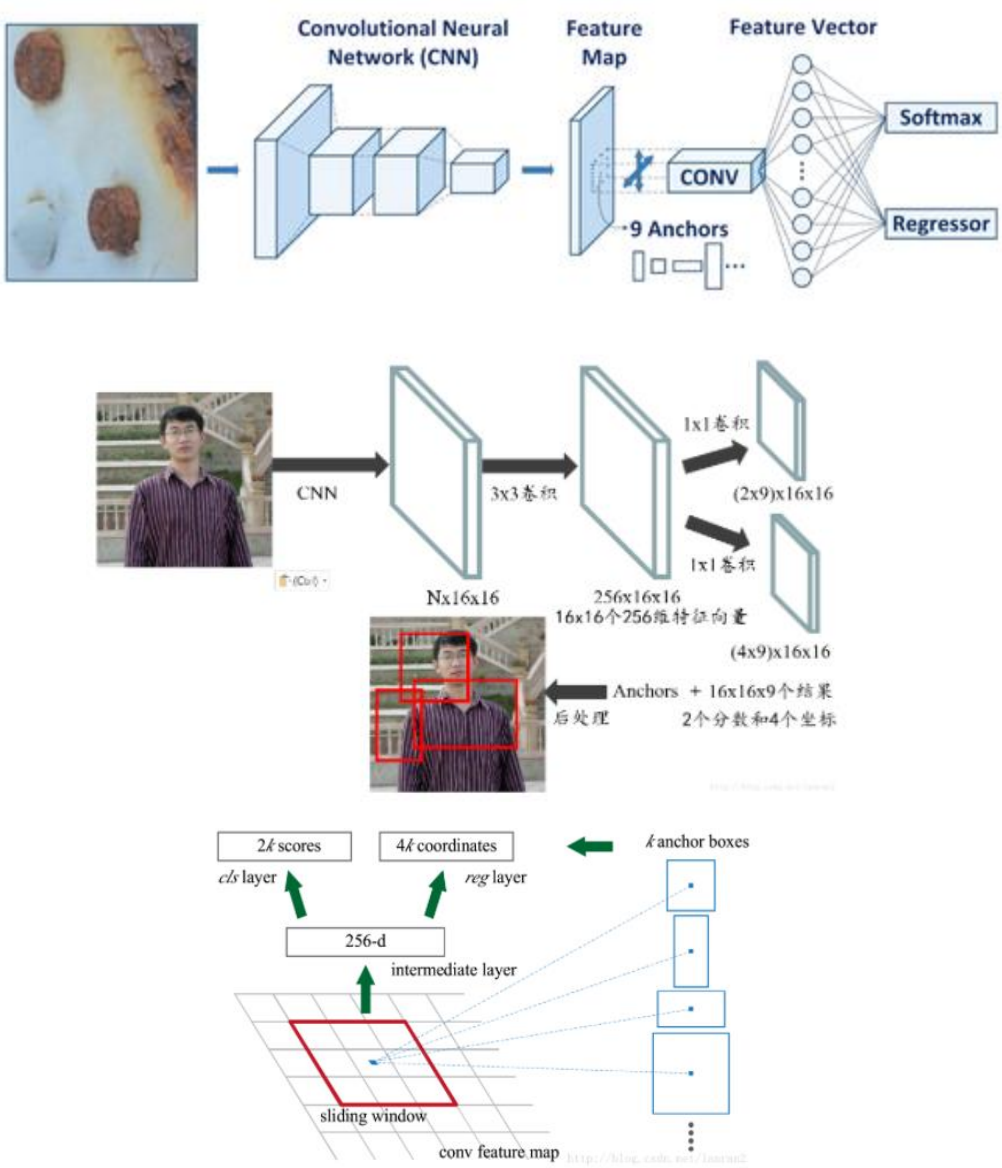
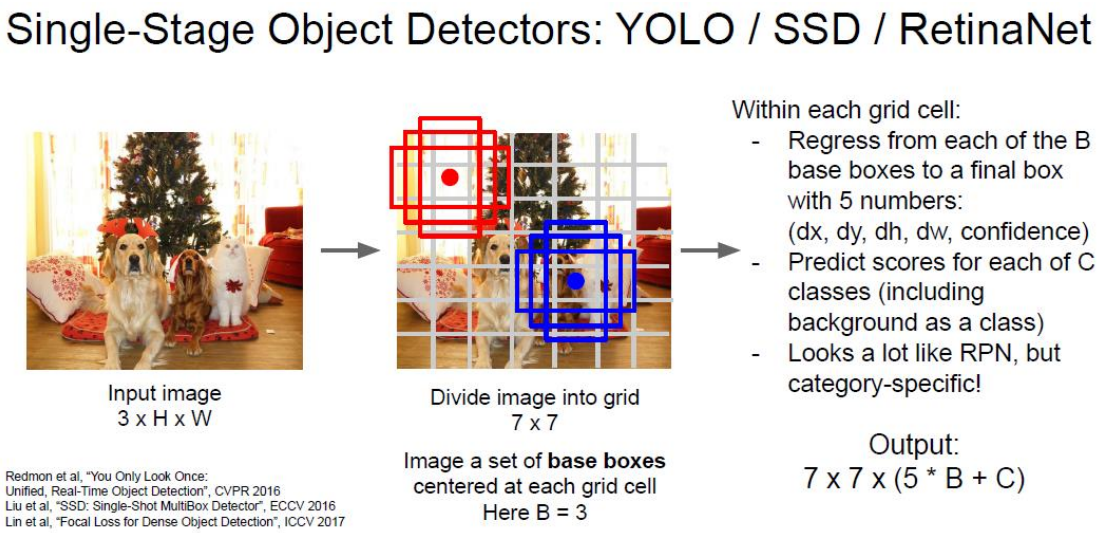


图2 RPN的结构

（3）基于单阶段的方法：YOLO、SSD…

利用整张图作为网络的输入，直接在输出层回归 bounding box(边界框)的位置及其所属的类别（详见笔记）

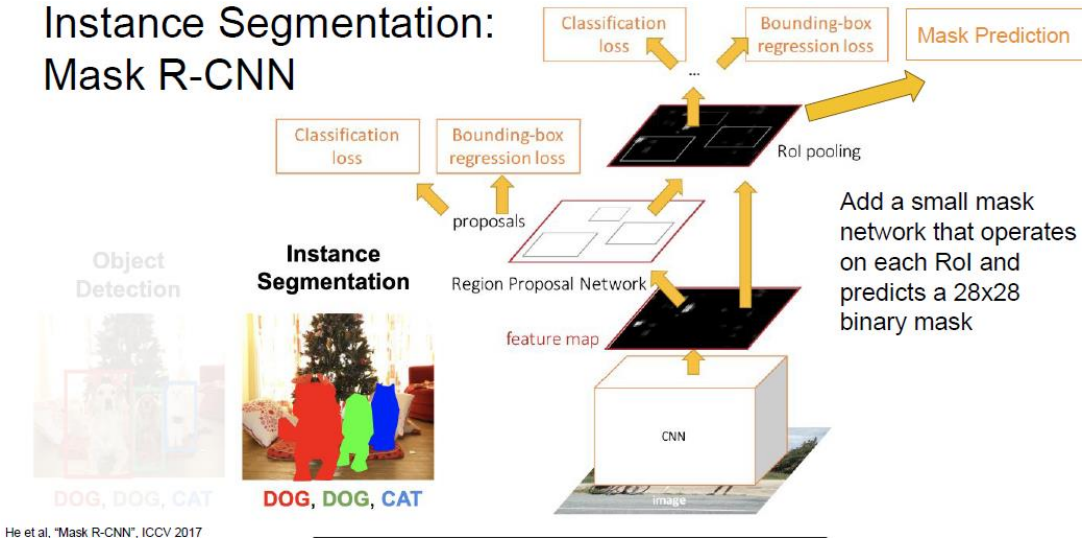


11.4 实例分割 Instance Segmentation

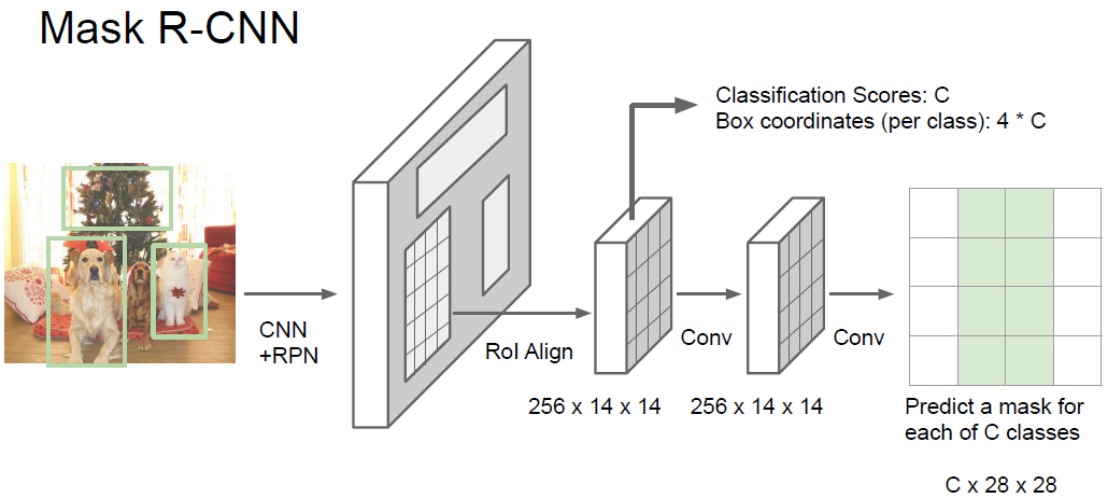
检测出输入图像中哪些像素点对应着检测目标（将图片中的每个目标检测出来，并得到这些目标对应的像素点）。

（1）Mask R-CNN

在 Faster R-CNN 的基础上增加语义分割模型——语义分割+目标检测



在采用 Faster R-CNN 的方法提取得到 Region proposal，在对每个 Region proposal 进行目标分类和定位的基础上（图中上方），进行语义分割（图中下方），得到 Region proposal 中每个像素点的分类


Mask R-CNN
Classification Scores: C
Box coordinates (per class): 4 * C
CNN
+RPN
RoI Align
Conv
Conv
256 x 14 x 14
256 x 14 x 14
Predict a mask for
each of C classes
C x 28 x 28