SAICMOTOR

Rich feature hierarchies for accurate object detection and semantic segmentation

R-CNN

汇报人：丁健刚

自我介绍

公司

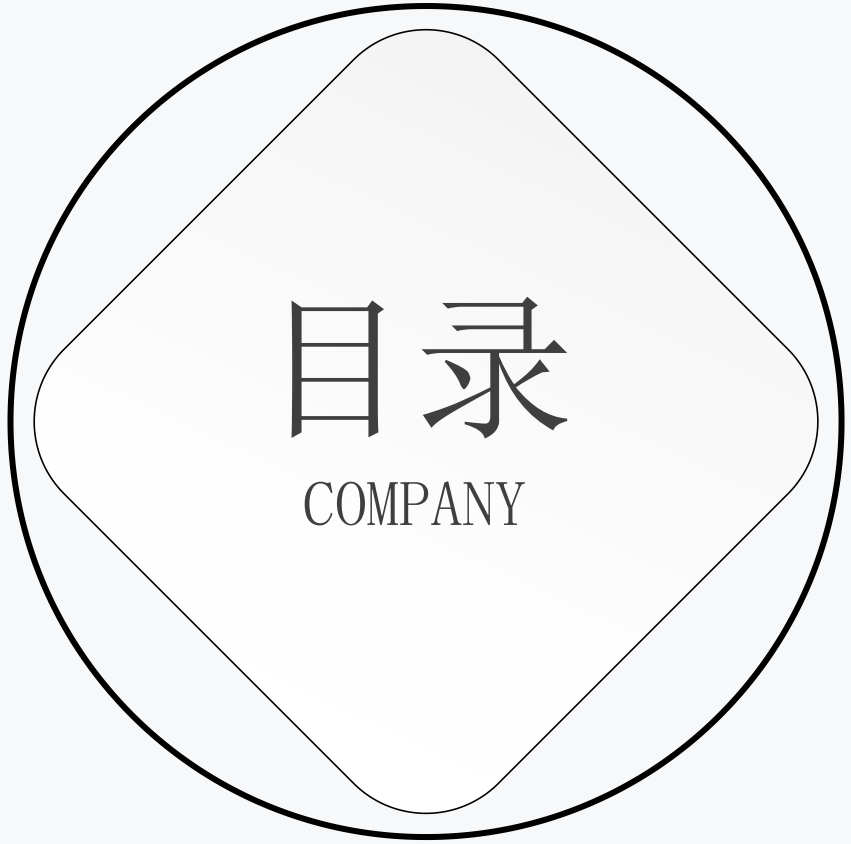
2014-2016 长城汽车哈弗技术中心 CAE部

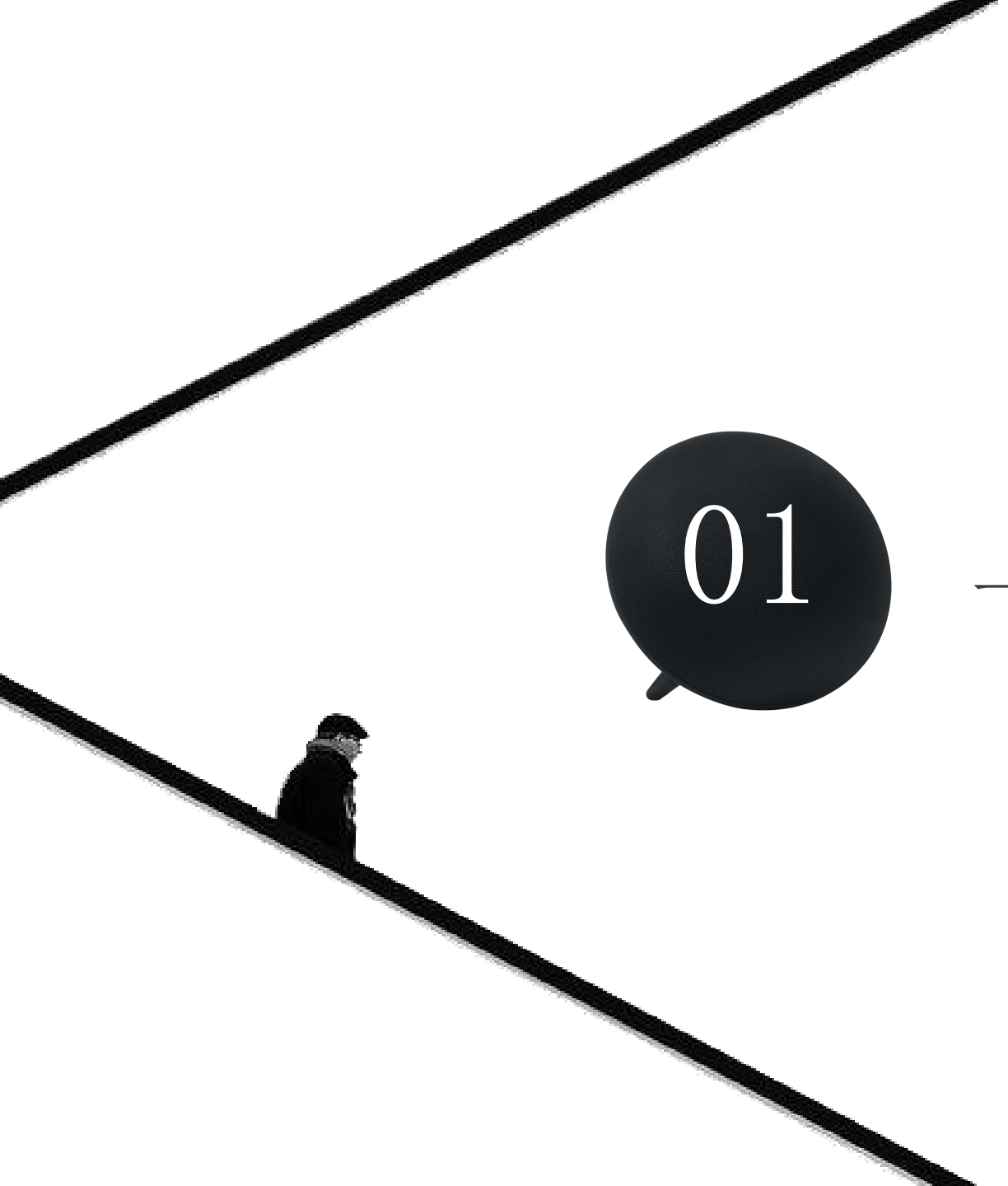
2016-至今 上汽商用车技术中心 智能驾驶部

专业

CAE 整车结构耐久性能分析

计算机视觉方向

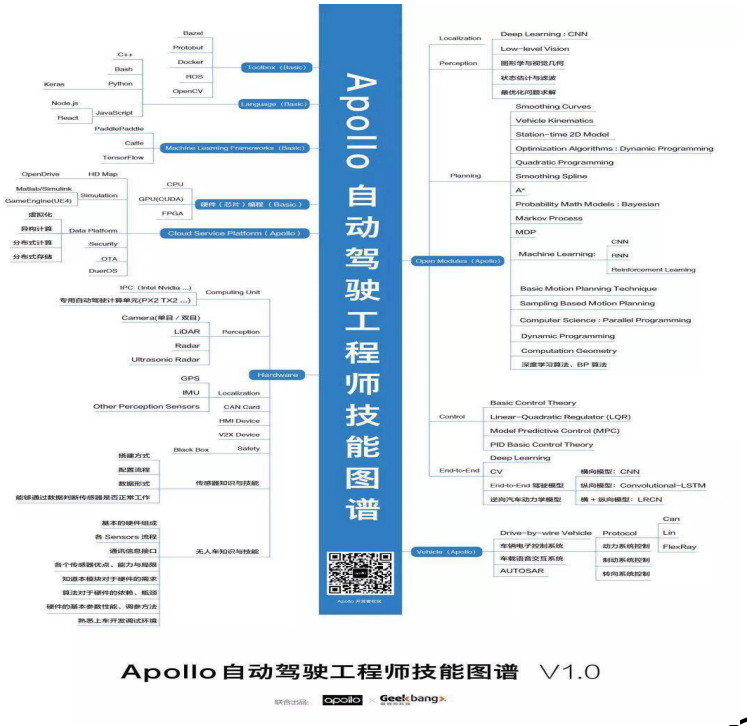




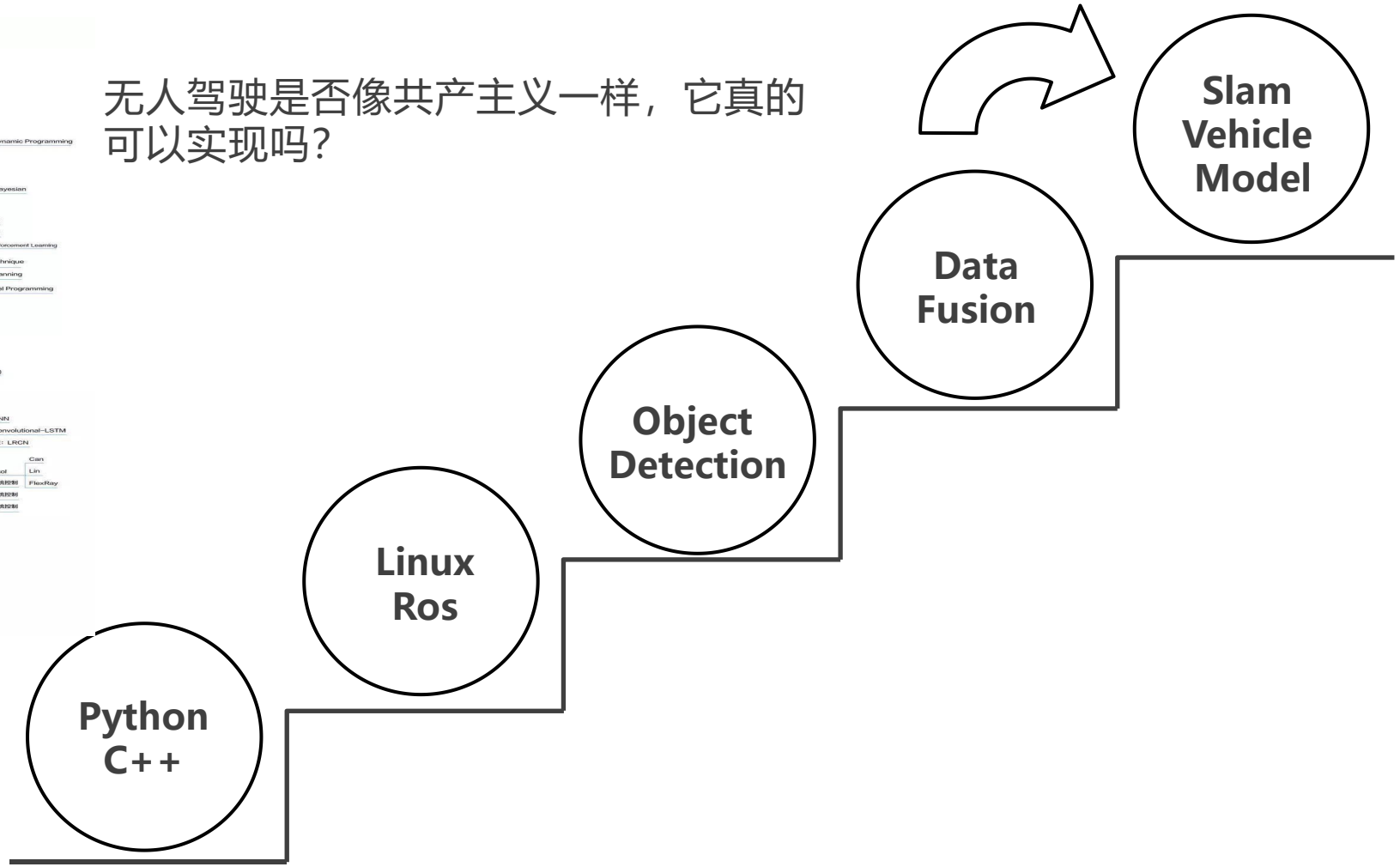
01 上车吧，少年

上车吧，少年

多传感器
摄像头/毫米波雷达/激光雷达。。。

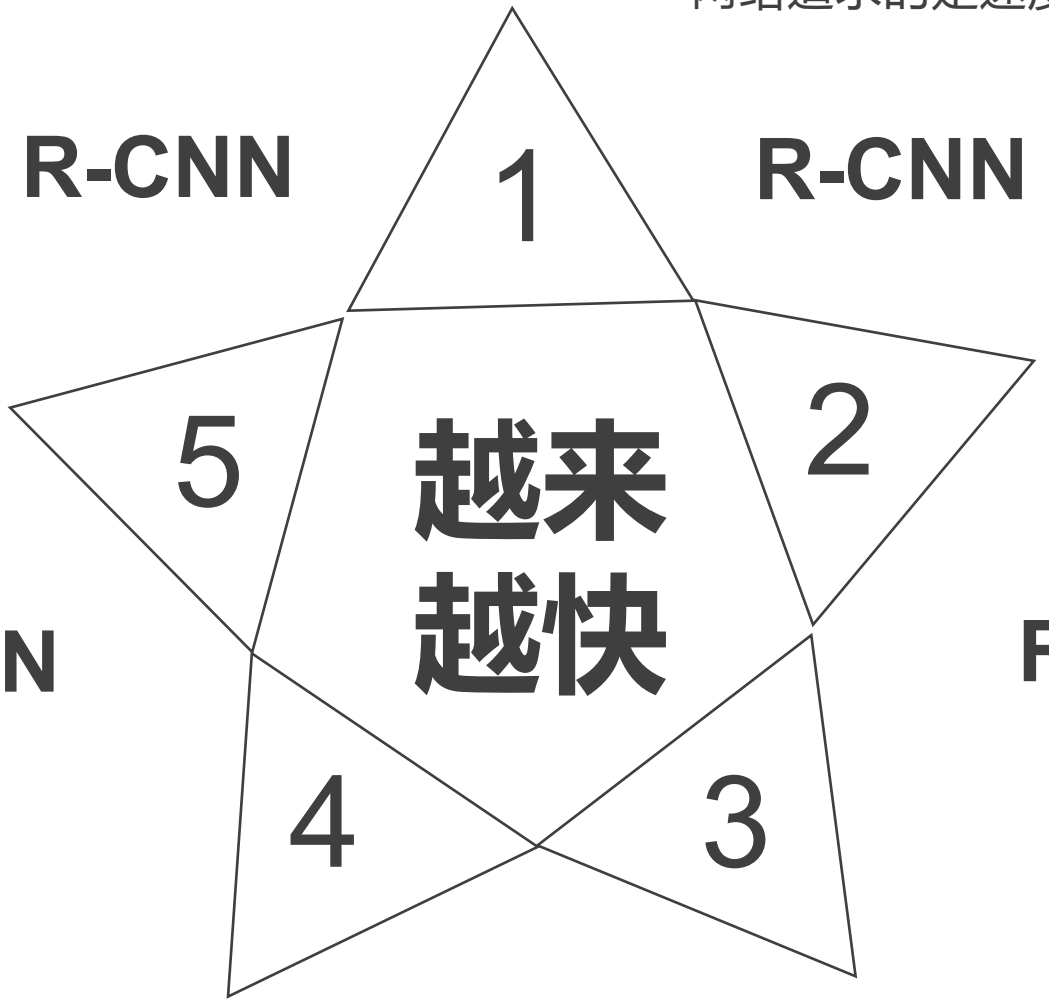


无人驾驶是否像共产主义一样，它真的可以实现吗？



论文框架与内容

追求什么
我们追求的是速度与激情
网络追求的是速度与精确度



R-CNN

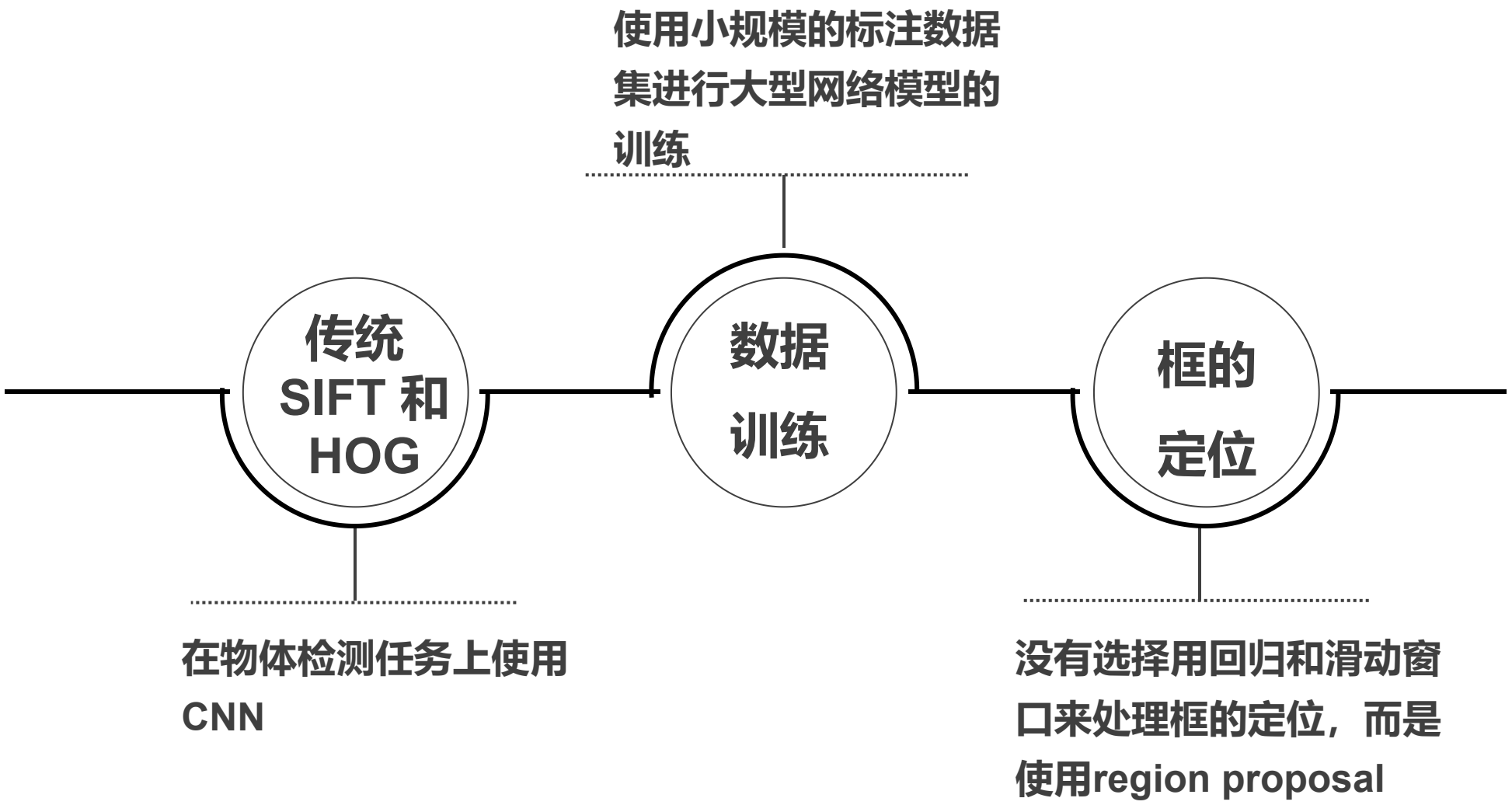
Fast R-CNN

Faster R-CNN

Light Head R-CNN

02 创新思路与方法

在目标检测中我们一直绕不开的话题



我们追求的是什么

01 如何处理数据，处理网络模型，防止过拟合，提高泛化能力

02 如何快速提取感兴趣的东西

03 如何提高识别精度与定位精度

04 如何提高架构的性能，追求实时准确

03 论文框架与内容

R-CNN基本框架

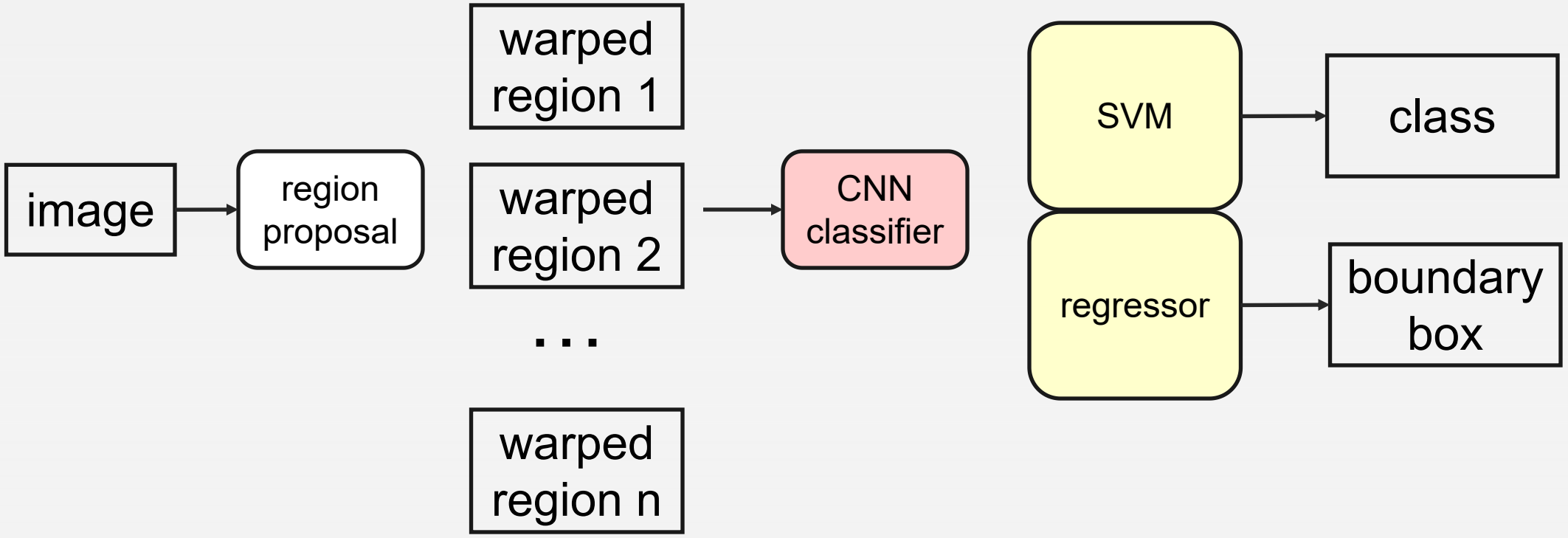


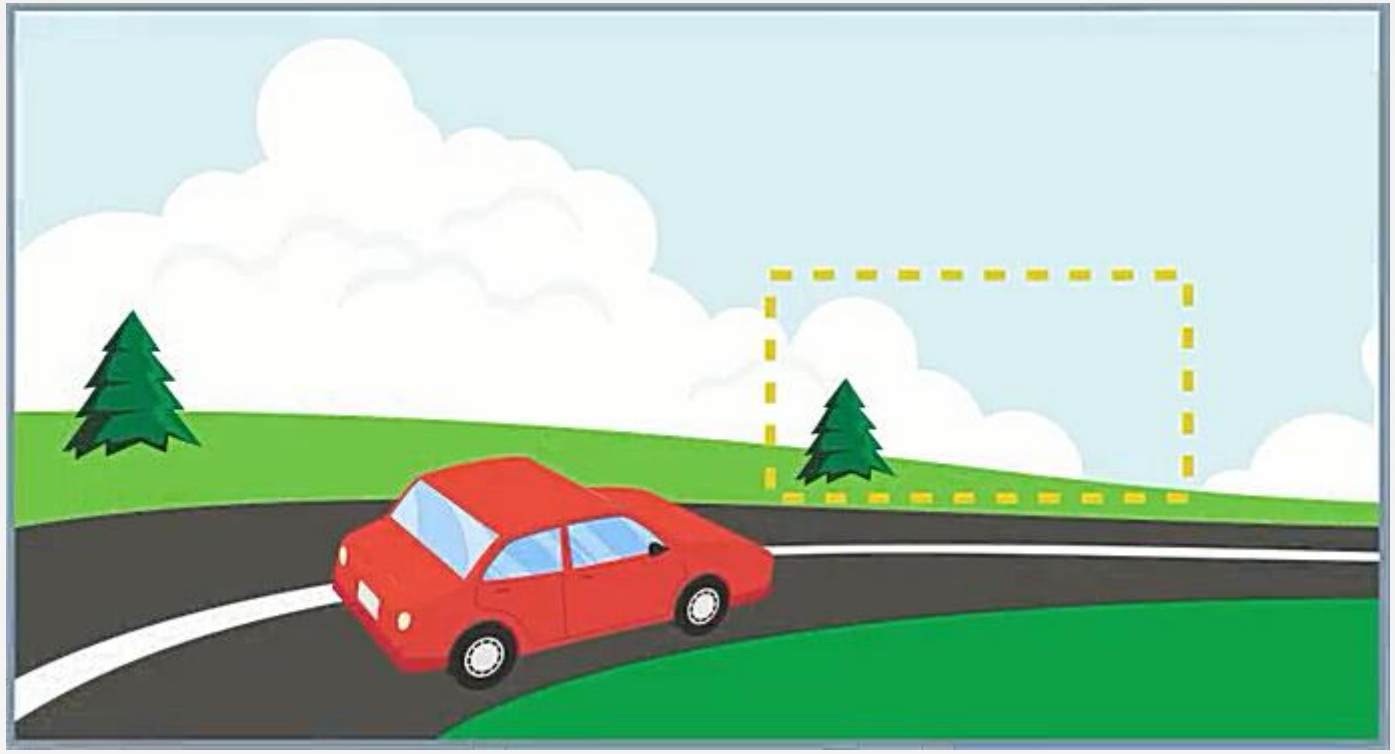
image
region proposal
warped region 1
warped region 2
...
warped region n
CNN classifier
SVM
class
regressor
boundary box

传统目标检测方法

SIFT & HOG

01 基于HOG的目标检测

特征描述了物体的特征，对于图像，它实际上归结为强度和强度的梯度
以及这些特征如何捕捉物体的颜色和形状，是图像信息的另一种数字表达



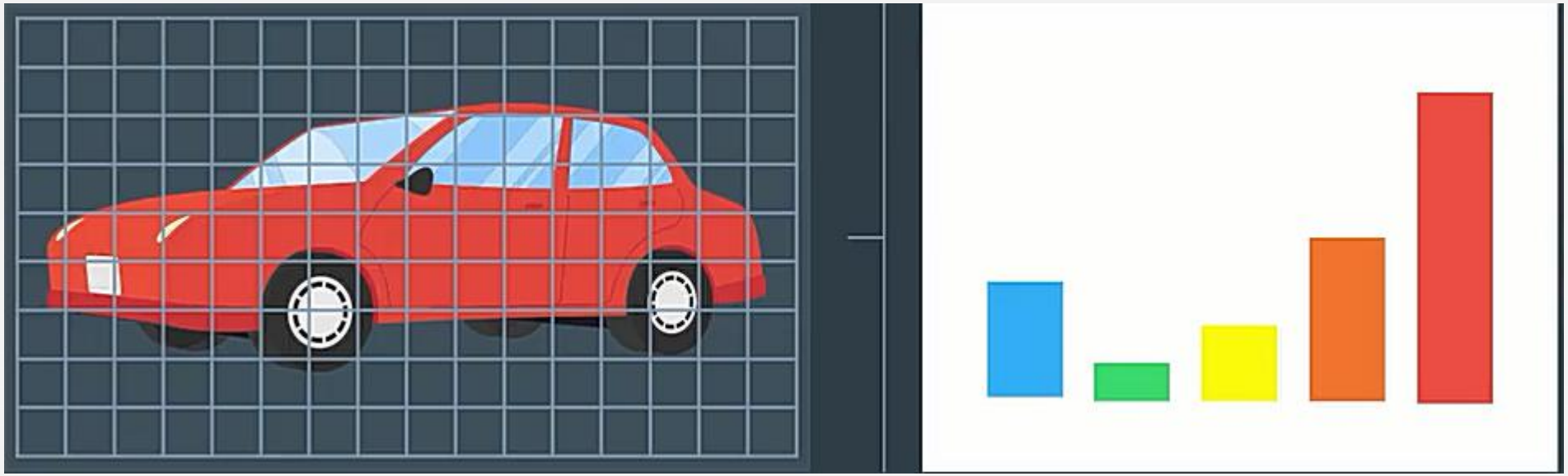
< thresh ?

RAW COLOR

Color Feature

传统目标检测方法

01 基于HOG的目标检测

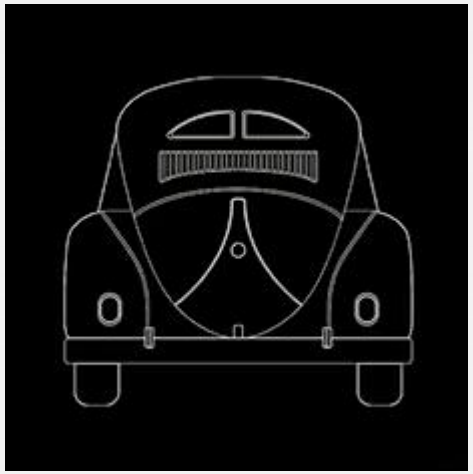


histogram

颜色空间LUV或HLS

传统目标检测方法

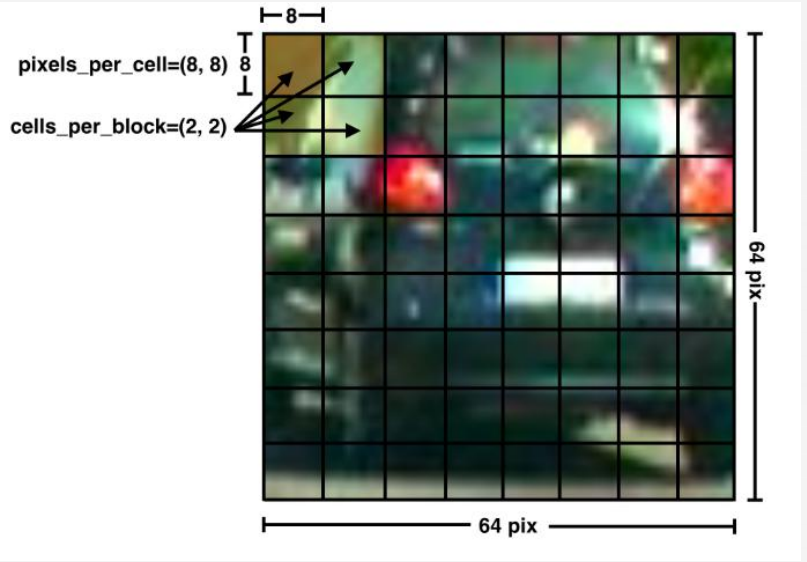
01 基于HOG的目标检测



但是如果汽车的颜色不同怎么办?
Gradient Features

scikit-image hog():

Histogram of Oriented Gradients(HOG)



传统目标检测方法

01 基于HOG的目标检测

Combining Feature

分类器

传统目标检测方法

02 SIFT

没办法，贴个百度概念：
SIFT，即尺度不变换特征，是用于图像处理领域的一种描述。这种描述具有尺度不变性，可在图像中检测出关键特征点，SIFT充分考虑了光照、尺度和旋转等变化，也带来了计算量的增加。

当我们谈及特征点时，是指提取关键点，并计算描述子这两件事。

如何找关键点

首先举个简单例子，当然这不是SIFT的方式

1.在图像中选区像素p，假设它的亮度为Ip
2.设置一个阈值T（比如Ip的20%）
3.以像素p为中心，选择半径为3的圆上的16个像素点

02 SIFT

如何找关键点

4.假如选区的圆上有连续N个点亮度大于Ip + T 或小于Ip – T，那么P就可以被认为时特征点

这里有好多特征点提取方法，FAST，SIFT，ORB，SURF

如何找描述子

其实也有好多方法，每种特征点提取方法的描述子计算都不太一样，所以无法穷尽了，也就举一个例子吧。

一种二进制描述子，其描述向量有许多个0和1组成，这里的0和1编码了关键点附近两个像素（比如p和q）的大小关系：如果p比q大，则取1，反之就取0。如果我们去了128个这样的p，q，最后就得到了128维由0和1组成的向量。

▶最开始做的永远是数据处理

各向同性和各向异性缩放

当我们输入一张图片时，我们要搜索出所有可能是物体的区域，R-CNN采用的就是Selective Search方法，通过这个算法我们搜索出2000个候选框。然后从R-CNN的总流程图中可以看到，搜出的候选框是矩形的，而且是大小各不相同。然而CNN对输入图片的大小是有固定的，如果把搜索到的矩形选框不做处理，就扔进CNN中，肯定不行。因此对于每个输入的候选框都需要缩放到固定的大小。

01

各向异性缩放

这种方法很简单，就是不管图片的长宽比例，管它是否扭曲，进行缩放就是了，全部缩放到CNN输入的大小227*227。附录A图D

02

各向同性缩放

因为图片扭曲后，估计会对后续CNN的训练精度有影响，于是作者也测试了“**各向同性缩放**”方案。有两种办法：

1. 先扩充后裁剪

直接在原始图片中，把bounding box的边界进行扩展延伸成正方形，然后再进行裁剪；如果已经延伸到了原始图片的外边界，那么就用bounding box中的颜色均值填充。附录A图B。

各向同性和各向异性缩放

02

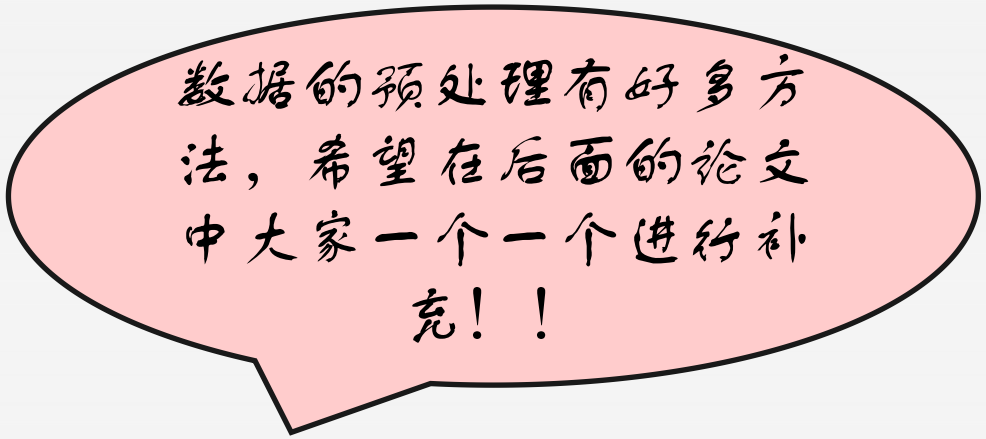
各向同性缩放

因为图片扭曲后，估计会对后续CNN的训练精度有影响，于是作者也测试了“**各向同性缩放**”方案。有两种办法：

2. 先裁剪后扩充

先把bounding box图片裁剪出来，然后用固定的背景颜色填充成正方形图片(背景颜色也是采用bounding box的像素颜色均值) 。附录A图C。

对于上面的异性、同性缩放，文献还有个padding处理，上面的示意图中第1、3行就是结合了padding=0, 第2、4行结果图采用padding=16的结果。经过最后的试验，作者发现采用各向异性缩放、padding=16的精度最高



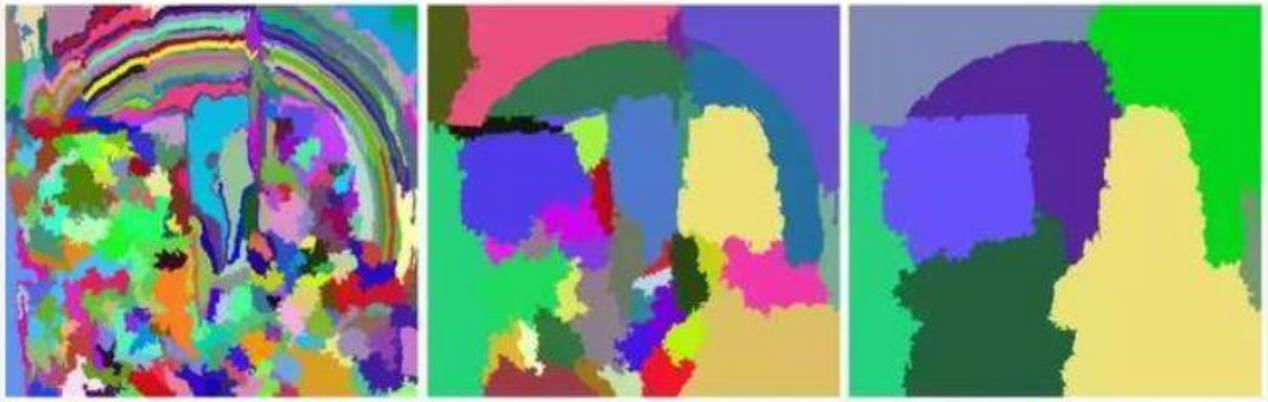
如何提取感兴趣的区域

基于候选区域的目标检测器

01 滑动窗口检测器 sliding window

暴力方法从左到右，从上到下...

02 选择性搜索 selective search



我们不使用暴力方法，而是用候选区域方法（region proposal method）创建目标检测的感兴趣区域（ROI）。在选择性搜索（selective search）中，我们首先将每个像素作为一组。然后，计算每一组的纹理，并将两个最接近的组结合起来。但是为了避免单个区域吞噬其他区域，我们首先对较小的组进行分组。我们继续合并区域，直到所有区域都结合在一起。

R-CNN 需要非常多的候选区域以提升准确度，但其实有很多区域是彼此重叠的，因此 R-CNN 的训练和推断速度非常慢。如果我们有 2000 个候选区域，且每一个都需要独立地馈送到 CNN 中，那么对于不同的 ROI，我们需要重复提取 2000 次特征。

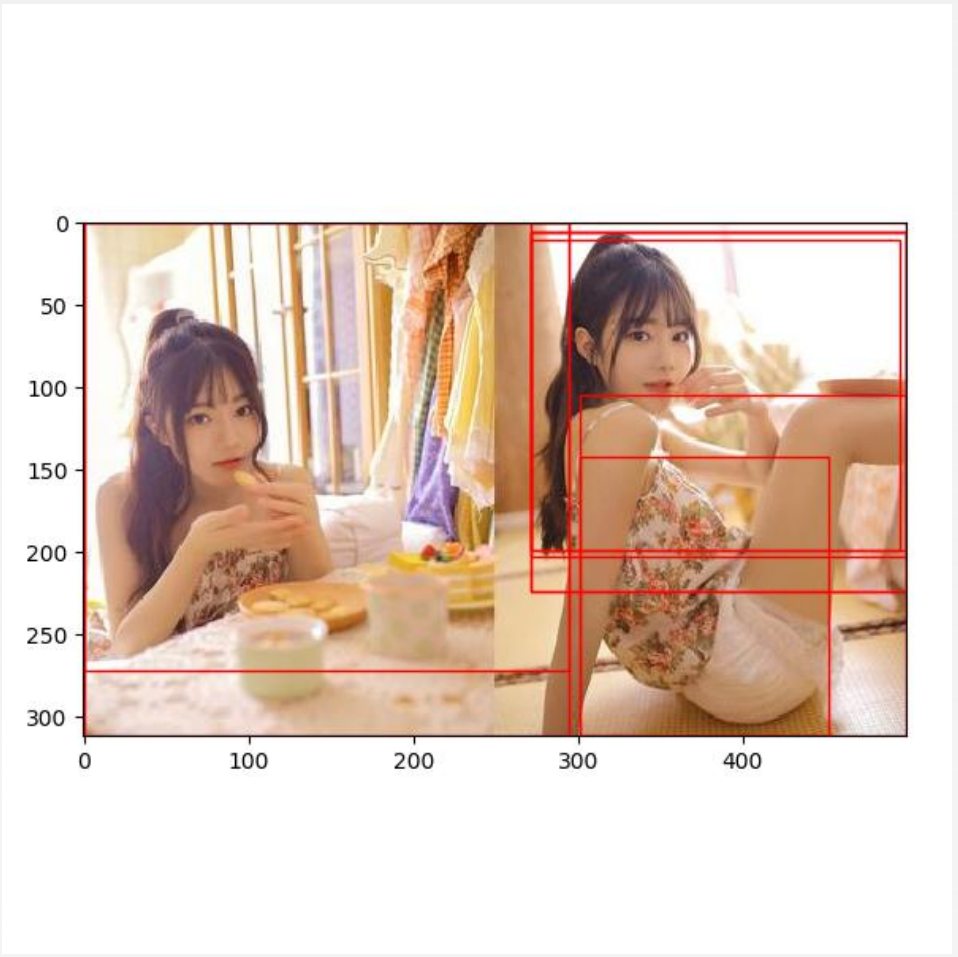
https://blog.csdn.net/Tomxiaodai/article/details/81412354

如何提取感兴趣的区域

理论联系实际

一个问题不太直观的时候，那么我们最好来点代码

先来个美女养养眼，就拿她做个例子



▶如何提取感兴趣的区域

https://github.com/AlpacaDB/selectivesearch

```
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import selectivesearch
from skimage import io
def main():
    img = io.imread('C:\\Users\\user\\Desktop\\11.jpg')
    img_lbl, regions = selectivesearch.selective_search(img, scale=500, sigma=0.9, min_size=10)
    candidates = set()
    for r in regions:
        if r['rect'] in candidates:
            continue
        if r['size'] < 2000:
            continue
        x, y, w, h = r['rect']
        if w / h > 1.2 or h / w > 1.2:
            continue
        candidates.add(r['rect'])
    fig, ax = plt.subplots(ncols=1, nrows=1, figsize=(6, 6))
    ax.imshow(img)
    for x, y, w, h in candidates:
        print(x, y, w, h)
        rect = mpatches.Rectangle((x, y), w, h, fill=False, edgecolor='red', linewidth=1)
        ax.add_patch(rect)
    plt.show()
if __name__ == "__main__":
    main()
```

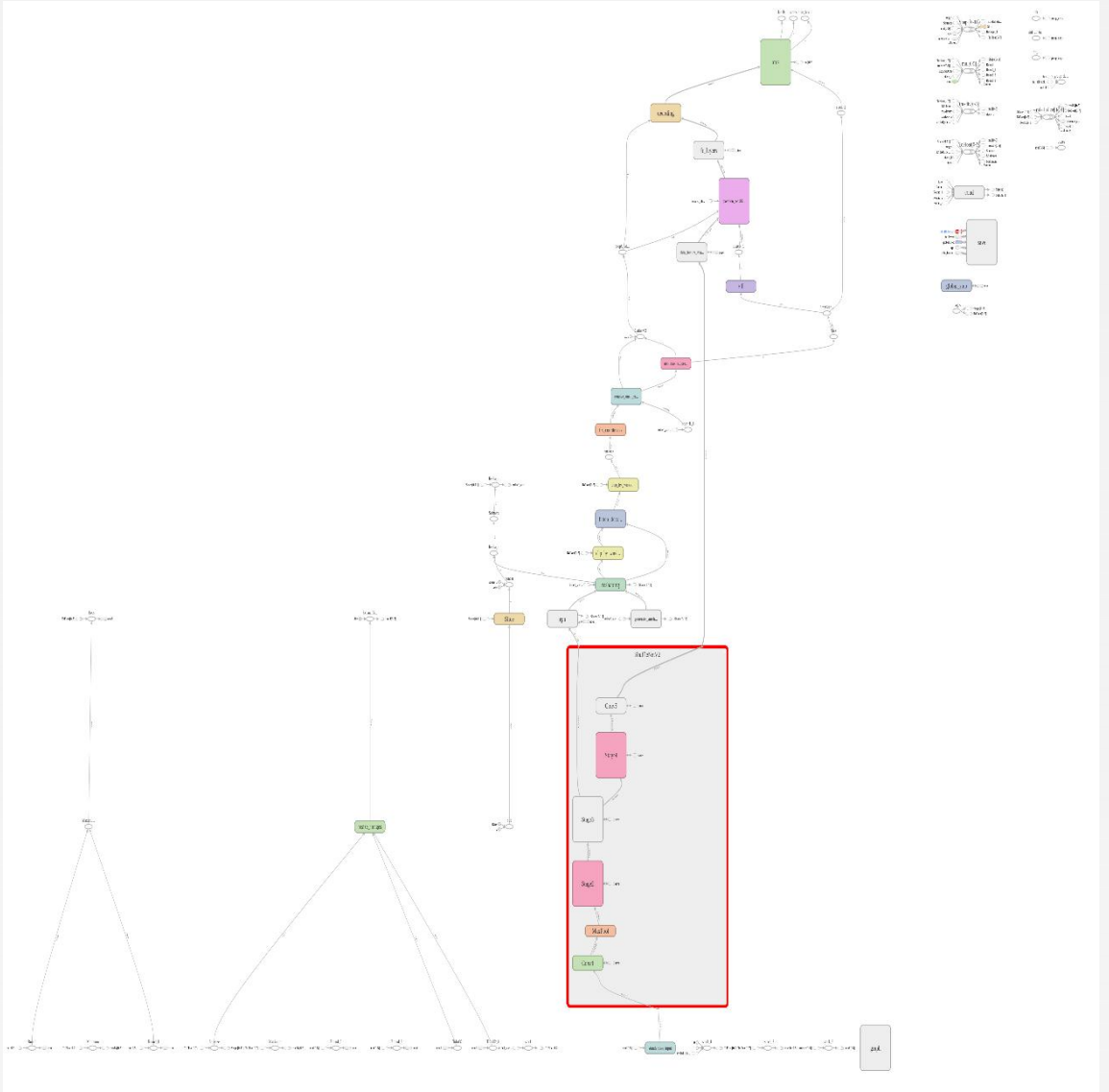
如何提取感兴趣的区域

03

RPN

基于候选区域的目标检测器

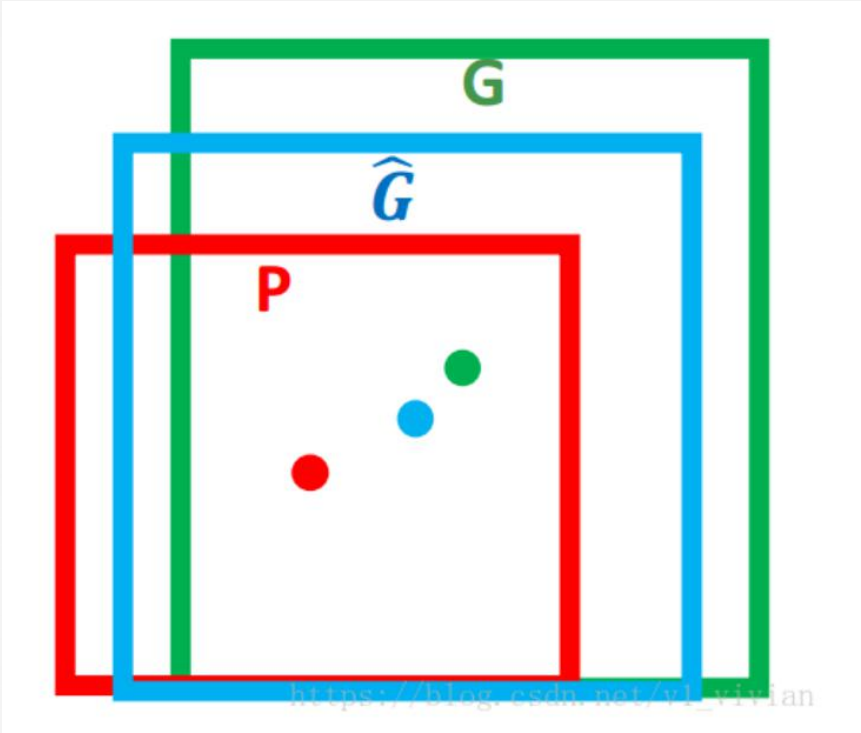
使用特征提取器（CNN）先提取整个图像特征，而不是从头对每个图像块提取多次



01

Bounding Box Regression

首先我们先来考虑，RCNN中为什么要做Bounding Box-Regression? Bounding Box regression是RCNN中使用的边框回归方法，在RCNN的论文中，作者指出：主要的错误是源于mis localization。为了解决这个问题，作者使用了bounding box regression。 这个方法使得mAP提高了3到4个点



01

Bounding Box Regression

对于预测框P，我们有一个ground truth是G：当0.1< IOU < 0.5时出现重复，这种情况属于作者说的poor localization， 但注意：我们使用的并不是这样的框进行BBR，作者是用IOU>0.6的进行BBR,也就是IOU<0.6的Bounding Box会直接被舍弃，不进行BBR。这样做是为了满足线性转换的条件。否则会导致训练的回归模型不 work。当 P跟 G 离得较远，就是复杂的非线性问题了，此时用线性回归建模显然不合理。

为什么IOU较大，认为是线性变换?

Log函数明显不满足线性函数，但是为什么当proposal和ground truth相差较小的时候，就可以认为是一种线性变换，看这个公式：

$$lim_{x=0} log(1+x) = x$$

再看公式：

$$t_w = \log(G_w / P_w) = log(\frac{G_w + P_w - P_w}{P_w}) = log(1 + \frac{G_w - P_w}{P_w})$$

01

Bounding Box Regression

当且仅当Gw – Pw=0的时候，才会是线性函数，也就是宽度和高度必须近似相等。
线性回归就是给定输入的特征向量 X, 学习一组参数 W, 使得经过线性回归后的值跟真实值 Y(Ground Truth)非常接近. 即Y≈WX 。
例如上图：我们现在要讲P框进行BBR，gt为G框，那么我们希望经过变换之后，P框能接近G框（比如，上图的G^框）。现在进行变换,过程如下：
我们用一个四维向量（x,y,w,h）来表示一个窗口，其中x,y,w,h分别代表框的中心点的坐标以及宽，高。我们要从P得到G^，需要经过平移和缩放。

平移公式：

$$\hat{G}_x = P_x + \Delta x$$

$$\hat{G}_y = P_y + \Delta y$$

缩放公式：

$$\hat{G}_w = P_w * \Delta w$$

$$\hat{G}_h = P_h * \Delta h$$

定位回归

01

Bounding Box Regression

△ 的表示：

$$\Delta x = P_w d_x(P)$$

$$\Delta y = P_h d_y(P)$$

$$\Delta w = e^{d_w(P)}$$

$$\Delta h = e^{d_h(P)}$$

变换的一般形式：

$$\hat{G}_x = P_x + P_w d_x(P)$$

$$\hat{G}_y = P_y + P_h d_y(P)$$

$$\hat{G}_w = P_w * e^{d_w(P)}$$

$$\hat{G}_h = P_h * e^{d_h(P)}$$

01

Bounding Box Regression

其实这并不是真正的BBR，因为我们只是把P映射回G^，得到一个一般变换的式子，那为什么不映射回最优答案G呢？于是，P映射回G而不是G^，那我们就能得到最优变换（这才是最终的BBR）：

$$G_x = P_x + P_w t_x(P)$$

$$G_y = P_y + P_h t_y(P)$$

$$G_w = P_w * e^{t_w(P)}$$

$$G_h = P_h * e^{t_h(P)}$$

01

Bounding Box Regression

这里为什么会将tw,th写成exp形式?
是因为tw,th代表着缩放的尺寸，这个尺寸是>0的，所以使用exp的形式正好满足这种约束。
也就是，我们将转换d换成转换t,就得到了P到G的映射。 di -> ti。 现在我们只需要学习
这四个变换dx(P),dy(P),dw(P),dh(P)，然后最小化t和d之间的距离，最小化这个loss，即可

注意：此时看起来我们只要输入P的四维向量，就可以学习,然后求出，但是，其实我们输入的是pool5之后的features，记做φ5，因为如果只是单纯的靠坐标回归的话，CNN根本就没有发挥任何作用，但其实，bb的位置应该有CNN计算得到的features来fine-tune。所以，我们选择将pool5的feature作为输入

$$d_i = w_i^T \Phi_{5i}$$

$$loss = \sum^{N} (t_i - w_i^T \Phi_{5i})^2 + \lambda ||w_i||^2$$

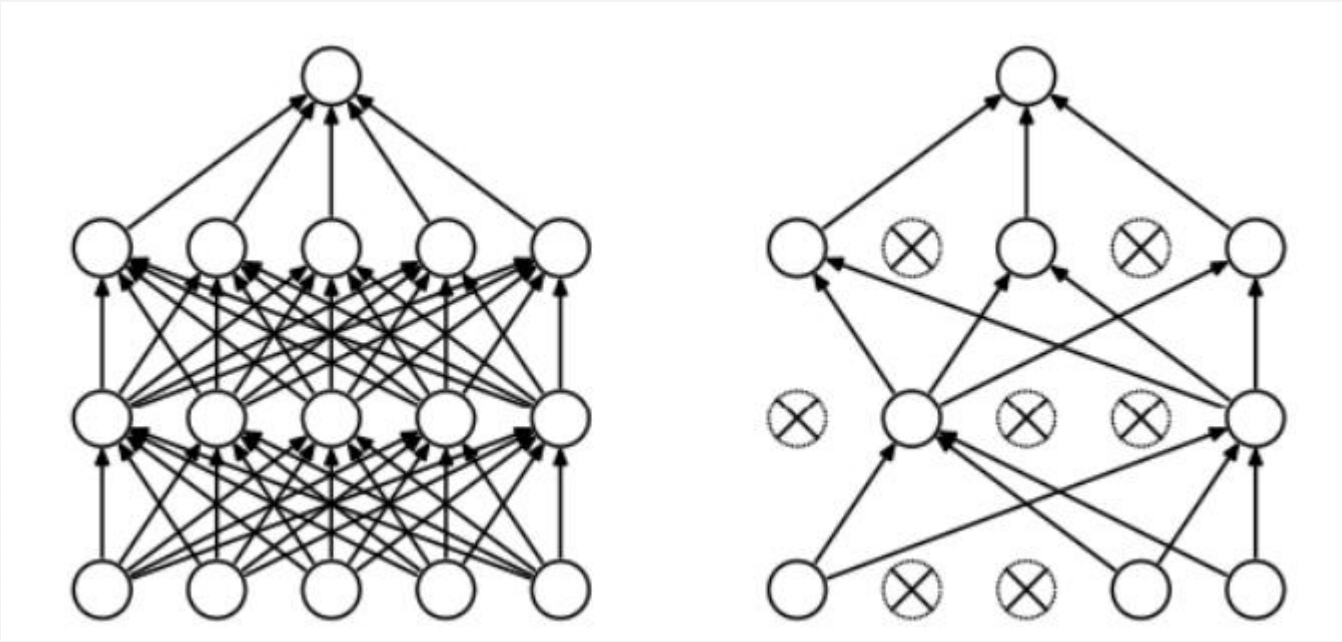
如何避免OverFitting

01

Dropout Regularization

Drop regularization

Dropout: A Simple Way to Prevent Neural Networks from Overfitting



如何避免OverFitting

Labeled Data is Scarce

01 Labeled data is scarce

传统方法：
先进行无监督训练，再进行有监督调优
本编论文核心贡献：
现在辅助数据集上进行有监督训练，再在小
数据集上针对特定问题调优

02 **Transfer Learning**

迁移学习涉及采用预先训练的神经网络并使神经网络适用新的不同数据集
取决于两者：
新数据集的大小和
新数据集与原始数据集的相似性

如何避免OverFitting

02

Labeled Data is Scarce

Transfer Learning

使用迁移学习的方法会有所不同，主要有四种情况：

- 数据集小，数据类似

切断神经网络的末端
添加一个新的完全连接的层，该层与新数据集中的类数相匹配
随机化新的完全连接层的权重; 冻结预训练网络中的所有权重
训练网络以更新新的完全连接层的权重

- 数据集小，数据不同

切断网络开始附近的大多数预训练层
向剩余的预训练层添加一个新的完全连接层，该层与新数据集中的类数相匹配
随机化新的完全连接层的权重; 冻结预训练网络中的所有权重
训练网络以更新新的完全连接层的权重

- 大数据集，数据类似

删除最后一个完全连接的图层，并替换为与新数据集中的类数相匹配的图层
随机初始化新的完全连接层中的权重
使用预先训练的权重初始化其余权重
重新训练整个神经网络

如何避免OverFitting

Labeled Data is Scarce

02

Transfer Learning

使用迁移学习的方法会有所不同，主要有四种情况：

- 大数据集，数据不同

删除最后一个完全连接的图层，并替换为与新数据集中的类数相匹配的图层
使用随机初始化的权重从头开始重新训练网络
或者，您可以使用与“大型和类似”数据案例相同的策略

如何避免OverFitting

01 参数调优

特定领域参数调优

- 只使用变形后的推荐区域对CNN参数进行SGD训练
- 替换掉imagenet的1000分类，换成了21分类，20类别数加一个背景
- 对所有推荐区域，如果其和真实标注的框的IOU>=0.5，认为是正例，否则就是负例
- SGD开始的learning rate为0.001
- 每轮SGD迭代，统一使用32个正例窗口和96个背景窗口，mini-bacth是128

- 霍夫曼概率投票

如何避免OverFitting

01

Hard negative mining method

对于目标检测中我们会事先标记处ground truth，然后再算法中会生成一系列proposal，这些proposal有跟标记的ground truth重合的也有没重合的，那么重合度（IOU）超过一定阈值（通常0.5）的则认定为是正样本，以下的则是负样本。然后扔进网络中训练。However，这也许会出现一个问题那就是正样本的数量远远小于负样本，这样训练出来的分类器的效果总是有限的，会出现许多false positive，把其中得分较高的这些false positive当做所谓的Hard negative，既然mining出了这些Hard negative，就把这些扔进网络再训练一次，从而加强分类器判别假阳性的能力

当我们搞不懂某些高大上的概念的时候，举个简单例子，可视化一下

01

可视化学习到的特征

- layer 1、layer 2学习到的特征基本上是颜色、边缘等低层特征；
- layer 3则开始稍微变得复杂，学习到的是纹理特征
- layer 4学习到的则是比较有区别性的特征，比如狗头
- layer 5学习到的则是具有辨别性关键特征

最终的最终还是要反复试验

01

消融研究（ablation studies)

ablation study 就是为了研究模型中所提出的一些结构是否有效而设计的实验。如你提出了某某结构，但是要想确定这个结构是否有利于最终的效果，那就要将去掉该结构的网络与加上该结构的网络所得到的结果进行对比，这就是ablation study。也就是（控制变量法）

RCNN实验结果证明：CNN的主要表达力来自于卷积层，而不是全连接层

SAICMOTOR

感谢观看 THANKS

汇报人：丁健刚