目标检测

吸取论文精华

R-CNN

R-CNN算法是深度神经网络在目标检测领域最早的实践，其主要过程分为四步：

- 利用Selective search算法在图像中生成1000-2000个候选区域
- 归一化为统一尺寸后送入卷积神经网络提取特征
- 神经网路输出的特征送入每一类对应的SVM二分类器分类
- 使用回归器精细修正候选框位置。

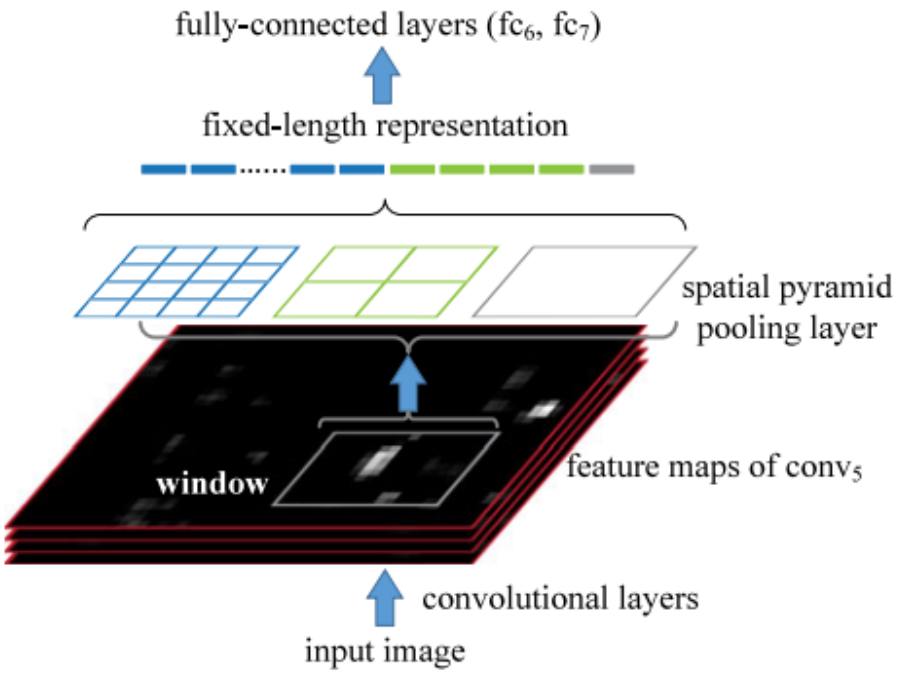
R-CNN的每个候选框都需要归一化后送到神经网路提取特征，计算量大。

SPP-net

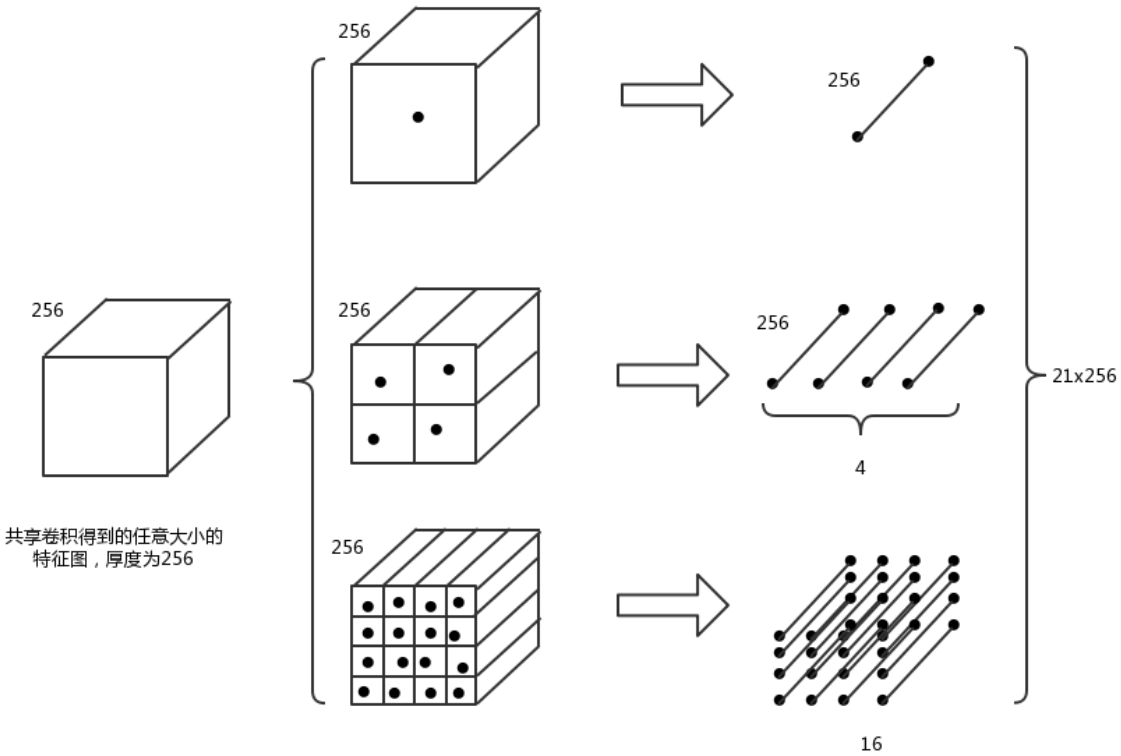
论文：Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition

SPPnet论文总结

本文在R-CNN的基础上思考卷积神经网络是否需要对输入fixed-size。卷积层、池化层对输入大小不敏感，输入不同尺寸对应输出不同大小；全连接层对输入尺寸有限制。所以归一化的过程可以放在全连接层前。而本文提出的归一化工具就是**Spatial Pyramid Pooling（SPP）**，即空间金字塔池化。SPP不仅在目标检测领域实用，而且可以广泛应用在计算机视觉各领域，作为fixed-size的一种解决方案。



基础网络最后卷积层输出的特征图（或者特征图的一部分）送入SPP；池化得到1*1，2*2，4*4的特征矩阵；拉平得到（1+4+16）的矩阵。下图展示了一个h*w*256的特征图转换为21*256的矩阵的SPP过程。SPP的巧妙之处在于巧用池化层将不确定大小的输入转变为相同大小的输出，优于暴力拉伸缩放。



Fast R-CNN

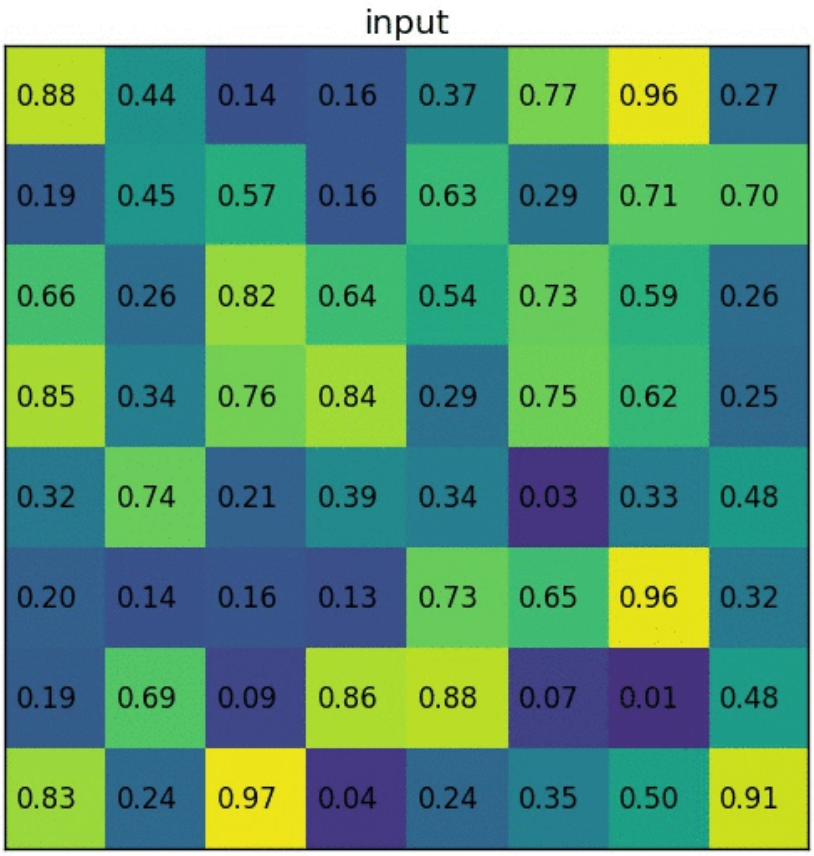
论文：Fast R-CNN

Fast R-CNN在R-CNN和SPP-net的基础上做了两点改进：

- 简化空间金字塔池化，提出**ROI pooling**（Region of interest pooling）
- 用卷积神经网络取代类别分类和位置回归，提出了多任务损失函数（即包括类别误差和位置误差）

ROI是表示预选框位置的五维矩阵（index, x1, y1, x2, y2），即图片索引和左上角右下角坐标。ROI pooling过程分为以下四步，下图展示不同尺寸的输入通过ROI pooling得到相同大小的特征图，：

1. 根据ROIs和特征图，得到候选框的信息
2. 将候选框分割为预设尺寸的小块
3. 寻找每个小块的最大值
4. 得到输出矩阵



YOLO

论文：You only look once: Unified, real-time object detection

以YOLO为代表的one-stage目标检测算法认为，单独的卷积神经网络就可以完成分类和定位两个任务。YOLO利用卷积层提取信息，全连接层输出类别与位置。检测过程将输入的图片划分为S*S个cell，每个cell负责预测中心点落在该cell的物体的类别和位置。一个cell预测B个bounding box，B个bbox共享类别概率，输出一组5*B+class_num维结果。以B=2，class_num=5为例，每个cell输出为：

$$[x_1, y_1, w_1, h_1, c_1, x_2, y_2, w_2, h_2, c_2, class_1 \ldots . class_5]$$

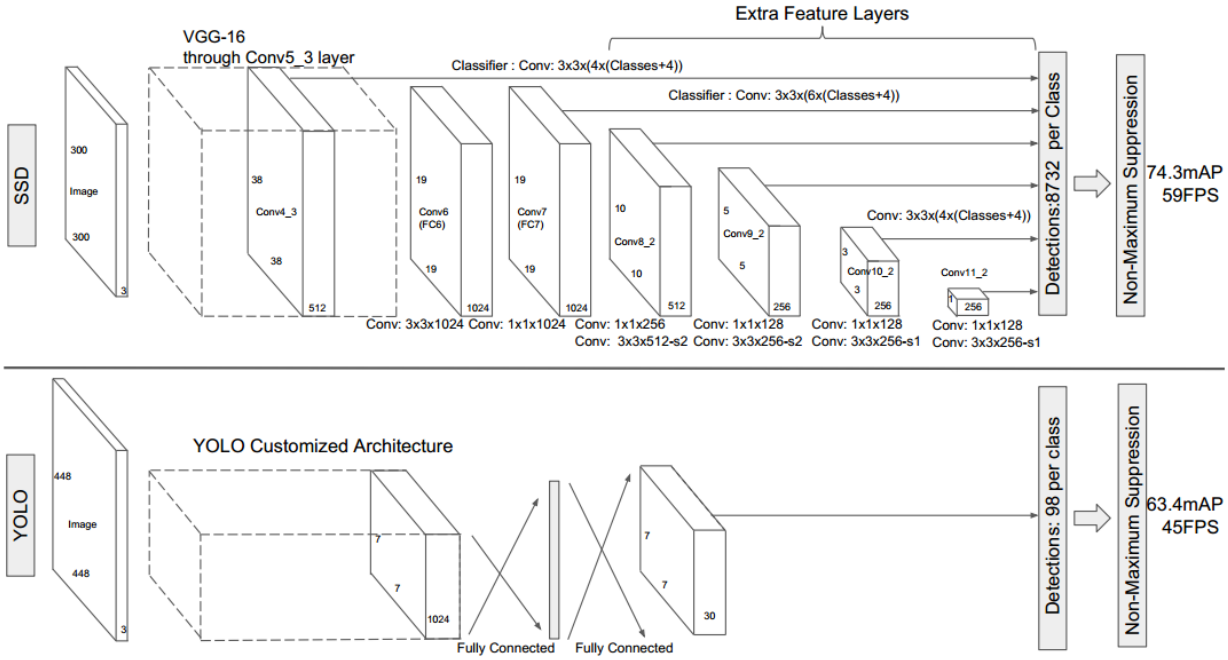
YOLO另一个创新是损失函数。传统的目标检测算法中，类别是分类问题，定位是回归问题；而YOLO将两者统一为回归问题，最近预测每个类别的概率而不是二分类。

SSD

论文：SSD: Single shot multibox detector

SSD与YOLO都采用单个神经网络实现分类定位。相对于YOLO，SSD作了一下改进：

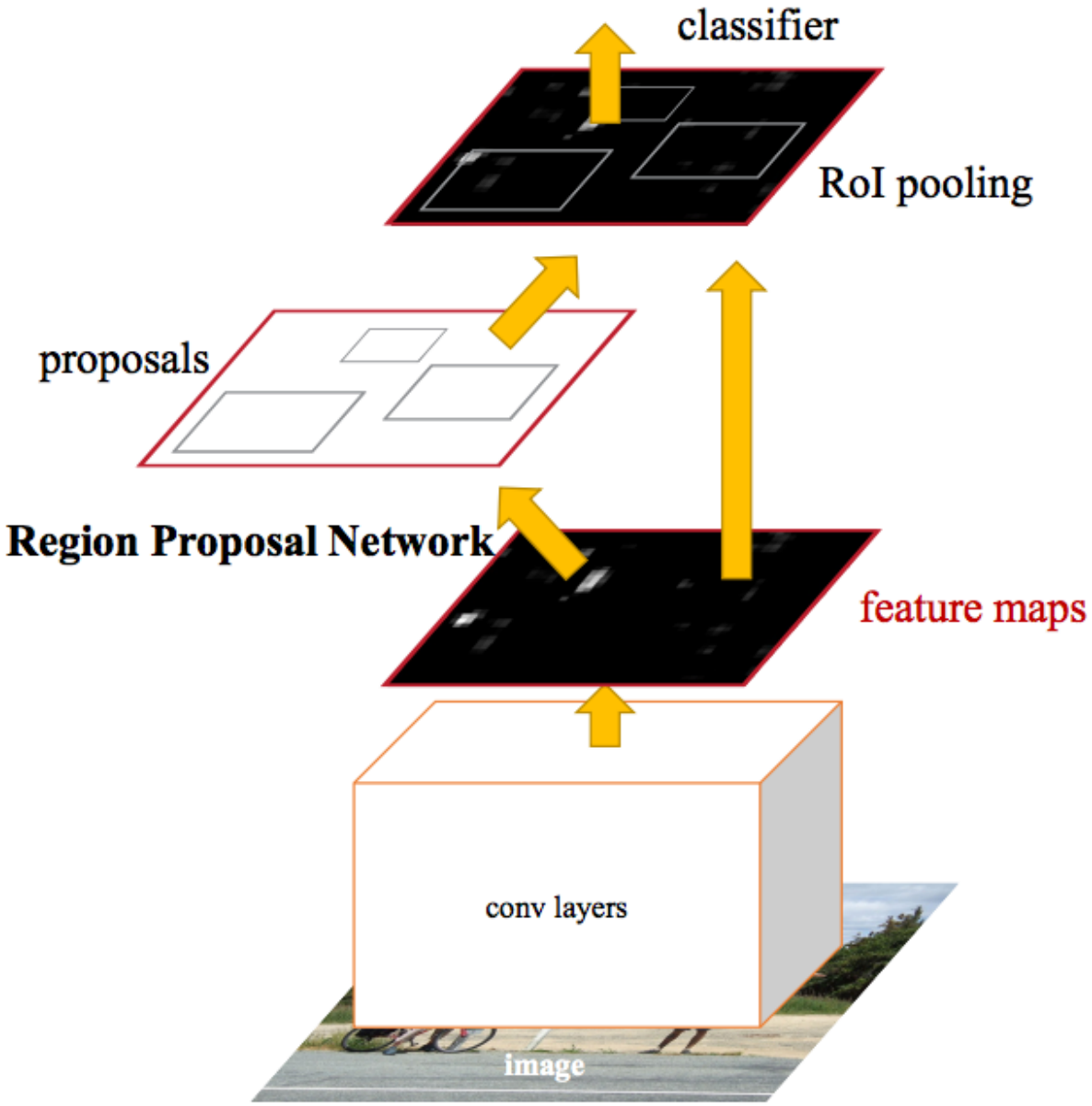
1. 借鉴ResNet的思路，将浅层的特征图连接到最后一层，显著提高了模型的准确度，特别是对小目标的识别能力；
2. 以Faster R-CNN的Anchor Box代替YOLO的Bounding Box；
3. 网络中部分使用DeepLab提出的空洞卷积。



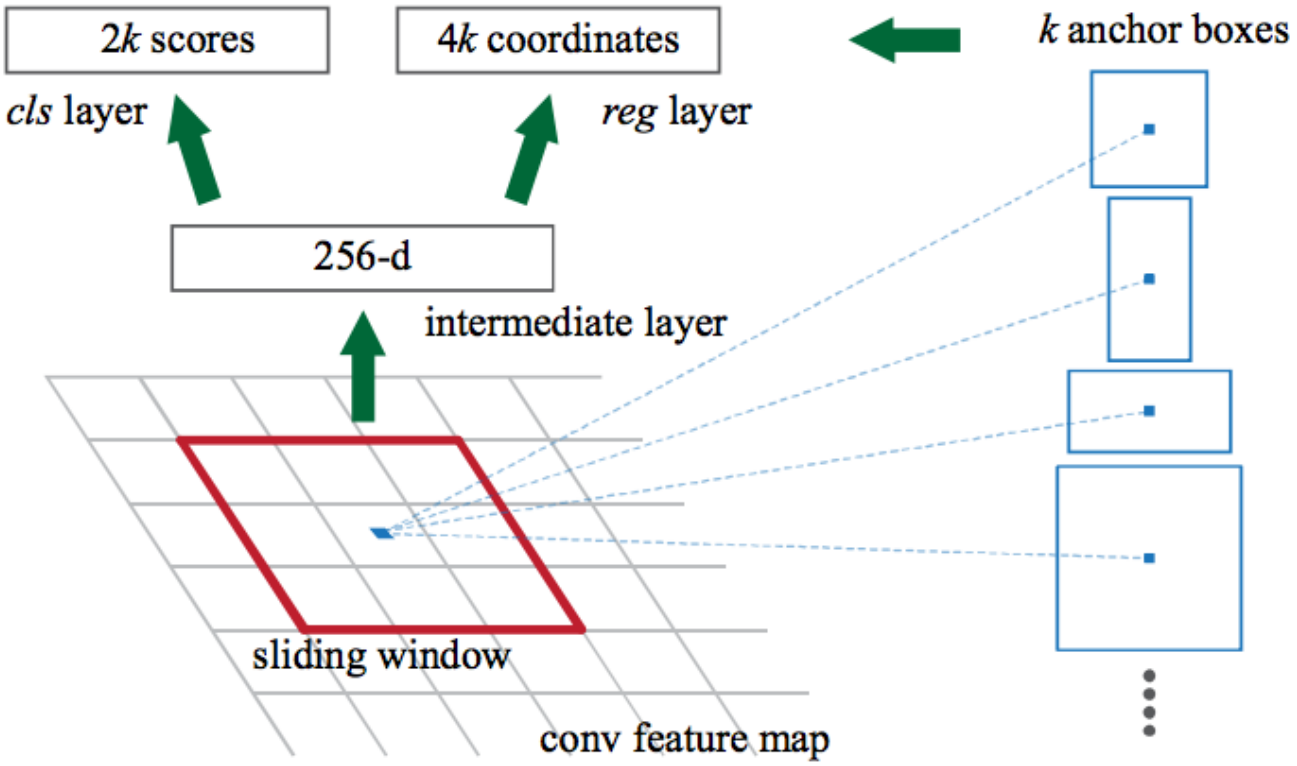
Faster R-CNN

论文：Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

从R-CNN升级到Fast R-CNN，整个算法还剩下一个瓶颈：生成候选框。Faster R-CNN放弃了传统候选框生成算法selective Search，设计了**RPN**（Region Proposal Network）算法，如图。



神经网络输出的特征图，送入RPN，就这么简单的得到候选框。RPN的输入与ROI pooling的输入相同，而且这两层均为单层卷积层（实际上还有1*1的卷积层），整个算法变成全卷积结构。故对于一张图片，只需要运行一次神经网络，节省大量计算开销。RPN生成候选框的过程如下：对于特征图的每个点，生成k个anchor boxes（一般设置3种scale和3种aspect rations，共9个anchor boxes）。每个anchor box预测6个参数，2个为存在物体和不存在物体的概率，另外4个是坐标。如果送入RPN的特征图尺寸为W*H，则预测W*H*k个anchor boxes，可以通过nms等方法过滤后送入ROI pooling。



YOLO v2v3

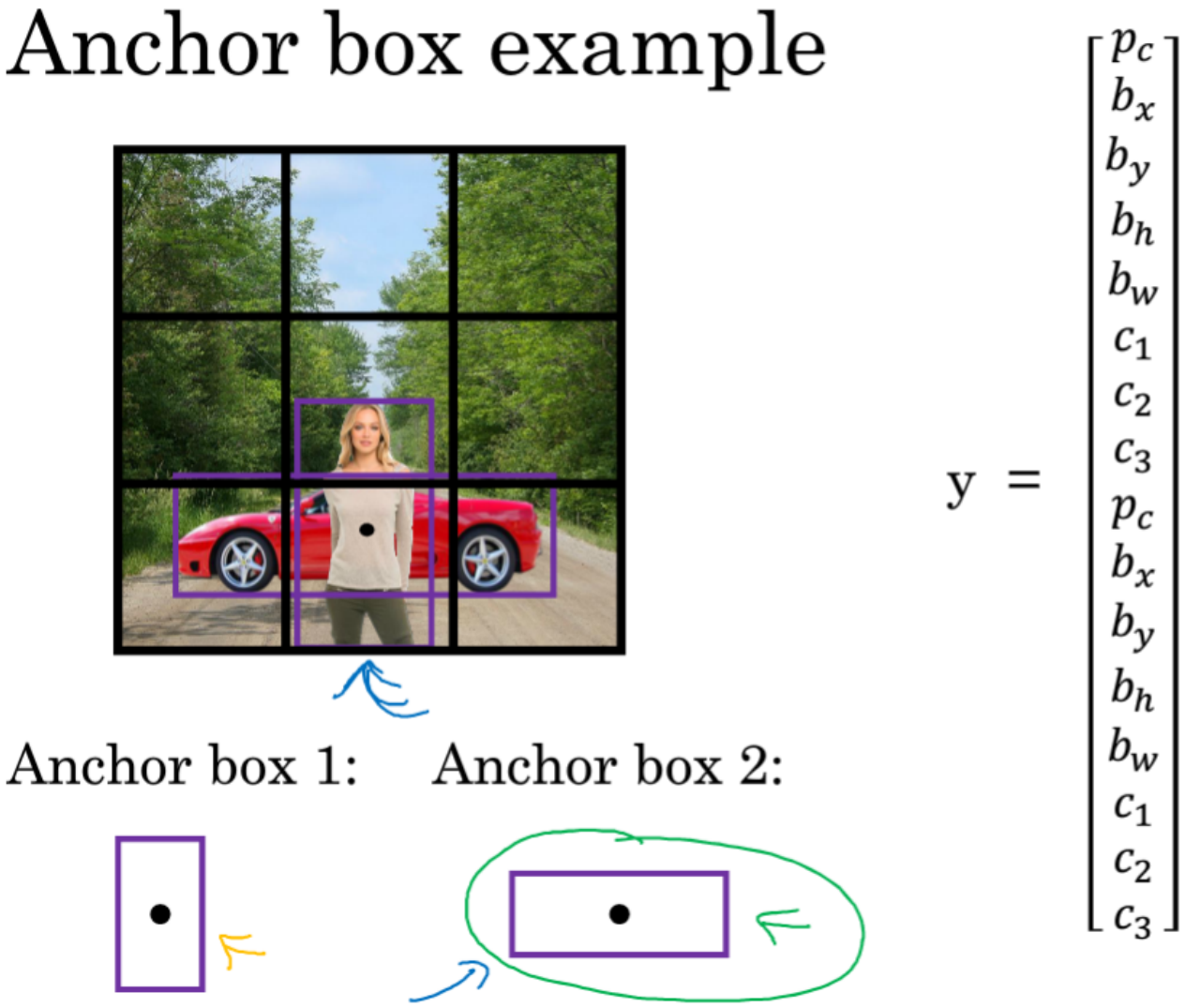
论文：YOLO9000: better, faster, stronger

anchor boxes

由于v1中，每个Cell的两个bbox共用一组类别概率 $class_1 . . . . class_n$，所以YOLO检测密集的物体易出错。所以，v2改进思路就是取消共用，让每个 bbox 都有自己的类别概率，如下：

$$[x_1, y_1, w_1, h_1, c_1, x_2, y_2, w_2, h_2, c_2, class_1 . . . . class_n] \rightarrow [x_1, y_1, w_1, h_1, c_1, class_1 . . . class_n], [x_2, y_2, w_2, h_2, c_2, class_1 . . . class_n]$$

除此之外，还引入Faster R-CNN中RPN（Region Proposal Networks），即anchor boxes，取代bbox。



实际操作中，通过预测偏移量 t_x, t_y, t_w, t_h，将预设的 anchor box 尺寸融入预测中。偏移量转换为 box 参数如下，在这组公式中 t_x, t_y, t_w, t_h 是 预测的box 的偏移量（预测值）； x, y, w, h 是预测的box 的中心点和宽高（由偏移量转换得到）； x_a, y_a, w_a, h_a 是 anchor boxes 的中心点和宽高（预先设定）。举个例子，如果 $t_x = 1$，相当于将 box 右移 w_a 。

$$
\begin{aligned}
t_x &= (x - x_a)/w_a \\
t_y &= (y - y_a)/h_a \\
t_w &= \log(w/w_a) \\
t_h &= \log(h/h_a) \\
\text{网络最终预测：} &[t_x, t_y, t_w, t_h, c, class_1 \ldots class_n]
\end{aligned}
$$

anchor boxes plus

论文 Dimension Clusters && Direct location prediction 部分

但是，这只是 anchor boxes 的初始形态。要让它 work，需要解决两个小问题：

- 选取合适的长宽

anchor boxes 需要设置 w，h，如何得到合适的 w和h 呢。Faster R-CNN 中没有提到如何选取 w h，本文提出一个有趣的方法：利用 kmeans 聚类得到 k 组长宽值。聚类过程：随机选择 k 个框为簇心，计算所有框到簇心的距离（定义距离为下式），将所有框分到与其距离最小的簇心，计算每簇的框的长宽均值为新的簇心，循环这个过程知道簇心不变或者到指定迭代次数。该论文通过实验，取 k = 5。

$$d(box, centroid) = 1 - iou(box, centroid)$$

- 保持训练的稳定

长宽设置好了，训练吧，又出问题了，模型不稳定，就是说不收敛或者是收敛有点慢。作者认为原因在于预测 x, y 上，网络预测 t_x, t_y 以表示 x, y ，但是 t_x, t_y 值可以取到整个图片，在训练的开始阶段，很难稳定。本文提出的解决方案是归一化，将 t_x, t_y, t_w, t_h, t_o 按照下式归一化：

$$g(z) = \frac{1}{1 + e^{-z}}$$

网络预测归一化后的 t_x, t_y, t_w, t_h, t_o , 参数转换方式为:

$$\begin{aligned} b_x &= \sigma(t_x) + c_x \\ b_y &= \sigma(t_y) + c_y \\ b_w &= p_w e^{t_w} \\ b_h &= p_h e^{t_h} \\ Pr(obiect) * IOU(b, object) &= \sigma(t_o) \end{aligned}$$

c_x, c_y 表示 cell 距离整张图片左上角的距离（可以看着是cell的左上角坐标）

至此，**YOLO v2 的 anchor boxes 介绍完毕。**v2 借鉴 Faster R-CNN 中的 anchor boxes, 并做以下几点改进：

1. 利用自定义的距离公式，kmeans 聚类得到合适的 anchor boxes 长宽值；
2. 利用归一化后的 t_x, t_y, t_w, t_h, t_o 训练，提高训练的稳定性。

YOLO 9000

除了算法上的改进，作者还在 v2 的基础上，别出心裁地训练了一个可以检测9000余种物体的模型，YOLO9000。训练一个可以检测 9000 种物体的模型难吗？不难，如果数据够的话。CV常用的数据集有coco和ImageNet数据集。coco数据集标签的种类不过百，但是每个物体都有精准的框数据；ImageNet数据集标签种类多，但是很多没有标框；coco数据集标签较为宽泛，比如coco有“狗”，而ImageNet有“金毛” “二哈”。所以，识别的数据足够，但是定位的数据不够。

作者提出一种联合训练的方法，混合coco和ImageNet数据集，利用coco训练网络定位，利用ImageNet训练网络分类。具体做法如下：将两个数据集混合作为训练集，并且得到如图所示的标签树，在“狗”大类中包含“二哈”和“金毛”小类；训练过程中，如果输入图片来自于coco数据集，则按照完整的 loss function 反向传播；如果输入图片来自 ImageNet，则仅采用 loss function 中的类别部分。这样，网络从coco中学习定位和大类的分类，从ImageNet中学习小类的分类。对于ImageNet中没有出现的某种狗，我们的模型可以把它归为狗这一大类，而不是不知道它是啥。

YOLO v3

- 增加了anchor box的数量
- 更加复杂的网络，53 层卷积层；
- 将 v2 中的 passthrough layer 发扬光大。

R-FCN

FPN

M2Det

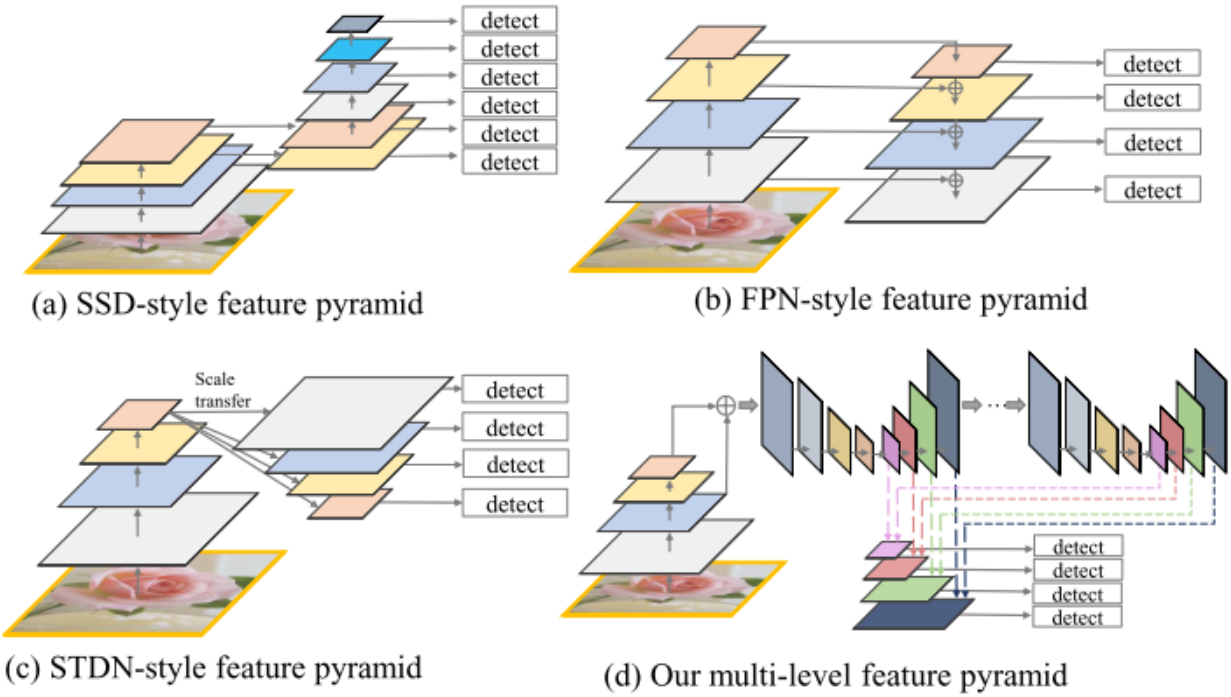
论文：M2Det: A Single-Shot Object Detector based on Multi-Level Feature Pyramid Network AAAI 2019

现有的 one-stage 和 two-stage 目标检测算法均广泛使用特征金字塔（feature pyramids），以解决物体间尺度变化带来的差异（scale variation across object instances）。

但是，现有的feature pyramids设计存在局限性：只是简单地根据内在多尺度构造金字塔结构，而且这些金字塔结构是被设计用于处理识别任务。本文首次提出多层次特征金字塔网络（Multi-Level Feature Pyramid Network，MLFPN），用于构建更加高效的特征金字塔，更好得解决目标检测任务中的多尺度问题。

首先，我们融合主干网络中的多尺度特征，得到基础特征；然后，将基础特征送入一组交替连接的简化U型模块和特征融合模块，每个简化U型模块中decoder layers输出一组多尺度特征图；最后，将多组多尺度特征图中的等尺寸特征组合，得到多层次特征金字塔，用于目标检测。为测试效果，将MLFPN集成到SSD中，得到M2Det。

Introduction and Related Work



目标检测现在面临的一个主要挑战是**物体间的尺度差异（Scale variation across object instances）**，通常采用两种策略解决：**image pyramid** 和 **feature pyramid**。image pyramid在测试时使用，会大大增加内存和计算复杂性，效率急剧下降。与image pyramid相比， feature pyramid占用的内存和计算成本更少，而且便于嵌入到各类现有的检测算法中。

尽管feature pyramid取得了不错的结果，但是其骨干网络的金字塔结构是被设计用于处理识别任务（如ResNet，虽然得到feature pyramid，但是存在缺陷，下文说明），所以根据这些骨干网络建立的特征金字塔存在一些局限性。例如，SSD直接单独使用两层骨架网络（VGG16）的特征，和通过步幅为2的卷积获得的四个额外层来构建特征金字塔；STDN仅使用DenseNet的最后一个Dense块，通过池化和尺度变换操作构建特征金字塔；FPN通过以自上而下的方式融合深层和浅层的特征来构造特征金字塔。

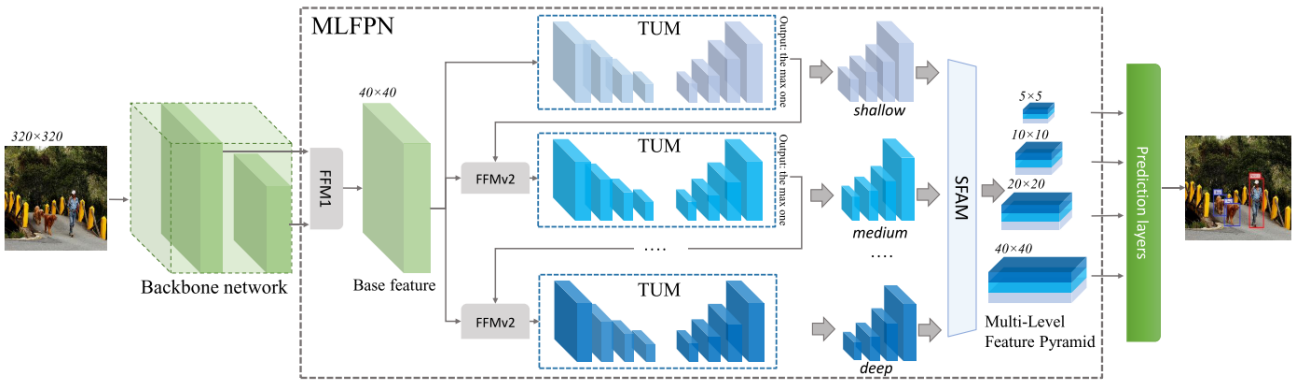
上述方法具有以下**两个限制。首先**，金字塔中的特征图对于目标检测任务而言不够典型（表达能力不够），它们只是简单地从为对象分类任务设计的骨干网络的层（特征）中构造。**其次**，金字塔中的每个特征图主要或甚至仅从骨干网络的单层构建，即主要或仅包含单层信息。本文分析总结了**各层特征图的特性**：较深层中的高级特征对分类任务更有效，而较浅层中的低级特征对定位任务更敏感。此外，低级特征更适合于表征具有简单外观的对象，而高级特征适合于具有复杂外观的对象。实际上，具有相似大小的对象实例的外观可能完全不同。例如，交通灯和遥远的人可能具有相当的尺寸，但是人的外观要复杂得多。因此，**feature pyramid性能欠佳的主要原因是金字塔中每个特征图主要由单级特征组成。**

本文构建一个更有效的feature pyramid，用于检测不同尺度的物体，同时避免上述局限。首先，我们融合主干网络中的多尺度特征，得到基础特征；然后，将基础特征送入一组交替连接的简化U型模块和特征融合模块，每个简化U型模块中decoder layers输出一组多尺度特征图；最后，将多组多尺度特征图中的等尺寸特征组合，得到多层次特征金字塔，用于目标检测。为测试效果，将MLFPN集成到SSD中，得到M2Det。

Proposed Method

M2Det利用基础网络和MLFPN提取输入图片的特征，得到密集的bounding boxes 和类别概率.。**MLFPN的作用就是将基础网络得到的多尺度特征叠加组合，得到新的多层次多尺度特征，以聚合浅层信息定位能力强、深层信息分类能力强的特点。**所以MLFPN模块可以作为独立组件拼接到各类目标检测网络中。MLFPN包含三个模块，FFM（Feature Fusion Module，特征融合模块），TUM（Thinned U-shape Module，简化U型模块）和SFAM（Scale-wise Feature Aggregation Module，多尺度特征增强模块）。

下图展示了M2Det_320 × 320的结构。在MLFPN中，FFMv1模块融合基础网络的特征图得到基础特征；TUM模块产生一组多尺度的特征（如图中shallow中的4尺度特征）；FFMv2模块融合基础特征和上一个TUM的最大尺寸特征图，送入下一个TUM；TUM和FFMv2模块交替连接，得到多层次多尺度（multi-level multi-scale）的特征（即图中shallow，medium，deep的3层特征）；最后，SFAM模块聚合多层次多尺度特征，得到新的融合的多尺度特征金字塔。level和scale都是参数，代码中取level=8，scale=6。



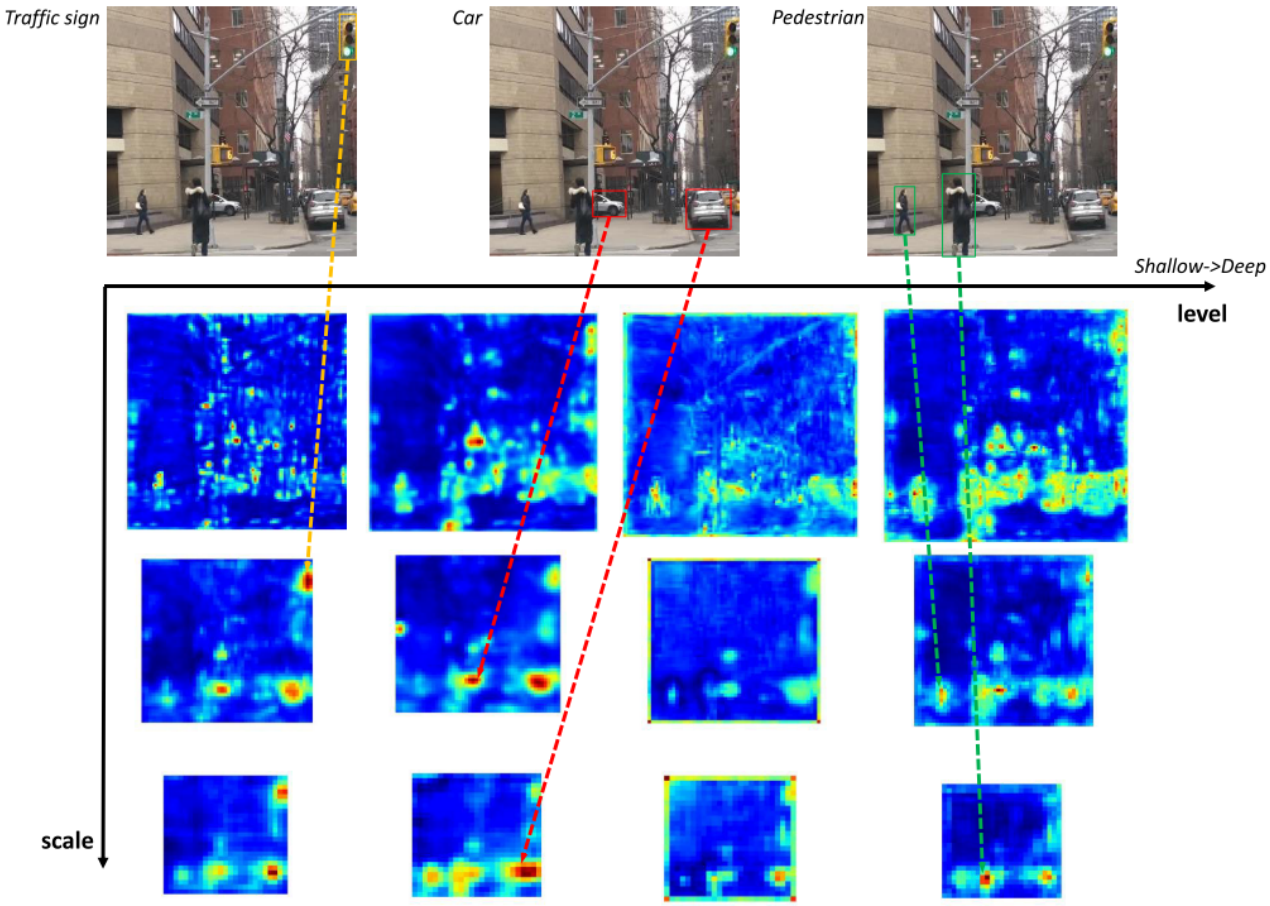
Discussion

作者还讨论了为什么MLFPN为什么有效。下图中，图片包含两个人，两辆车和一个指示灯。两个人之间，两辆车之间，大小不同；指示灯，小人小车的尺寸相近。通过比较不同层次不同尺度的特征图，可以得到三点：

- compared with the smaller person, the larger person has strongest activation value at the feature map of large scale, so as to the smaller car and larger car;
- the traffic light, the smaller person and the smaller car have strongest activation value at the feature maps of the same scale;
- the persons, the cars and the traffic light have strongest activation value at the highest-level, middle- level, lowest-level feature maps respectively.

这个例子表明：

- our method learns very effective features to handle scale variation and appearance-complexity variation across object instances;
- it is necessary to use multi-level features to detect objects with similar size.


Traffic sign
Car
Pedestrian
Shallow->Deep
level
scale