Object Detection

Contents

[RCNN 2](#_Toc90044976)

[2014 R-CNN 2](#_Toc90044977)

[2015 SPP-Net 4](#_Toc90044978)

[2015 Fast\_R-CNN 6](#_Toc90044979)

[2015 Faster\_R-CNN 9](#_Toc90044980)

[2016 R-FCN 12](#_Toc90044981)

[2017 Mask\_R-CNN 15](#_Toc90044982)

[Yolo 16](#_Toc90044983)

[2016 Yolo\_v1 16](#_Toc90044984)

[2017 Yolo\_v2 19](#_Toc90044985)

[2018 Yolo\_v3 21](#_Toc90044986)

[SSD 22](#_Toc90044987)

[2016 SSD 22](#_Toc90044988)

[2017 DSOD 24](#_Toc90044989)

[2017 DSSD 26](#_Toc90044990)

[2017 RSSD 29](#_Toc90044991)

[Other 31](#_Toc90044992)

[2013 OverFeat 31](#_Toc90044993)

[2017 FPN 32](#_Toc90044994)

[2017 RetinaNet & Focal loss 34](#_Toc90044995)

[Merge 36](#_Toc90044996)

[2017 RON 36](#_Toc90044997)

[2018 RefineDet 39](#_Toc90044998)

# RCNN

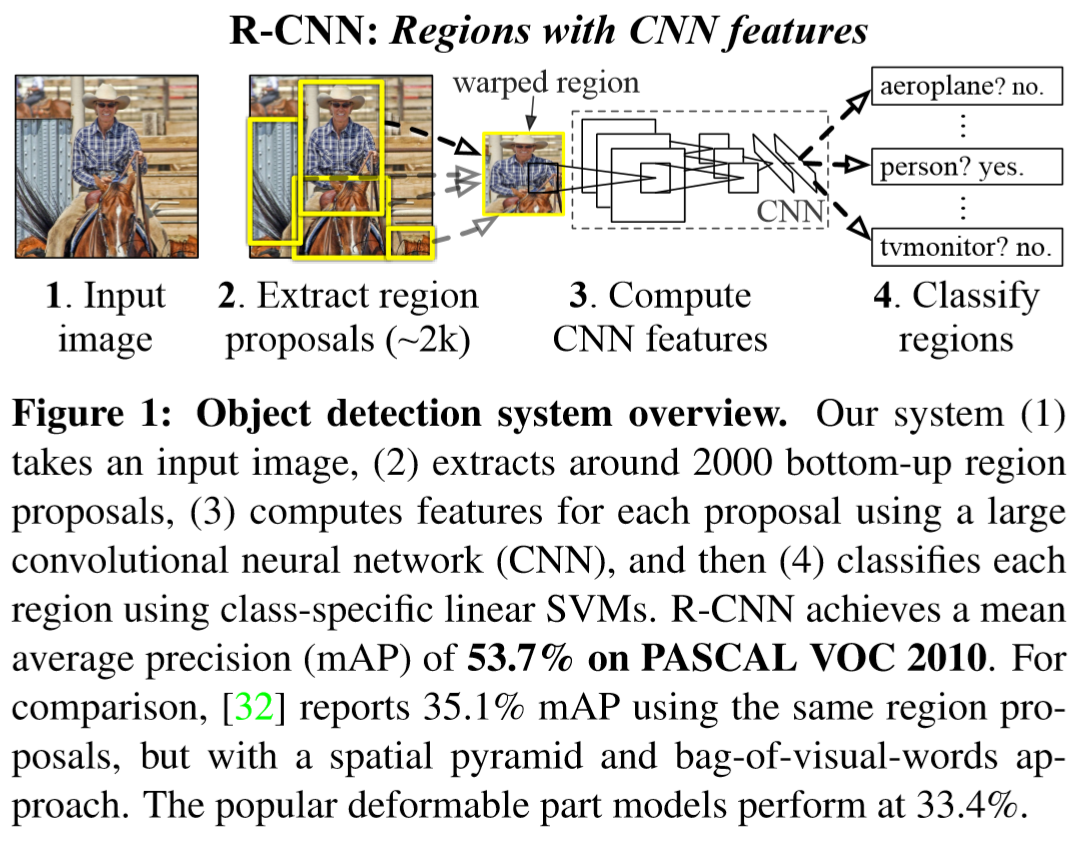
## 2014 R-CNN

CVPR - Rich feature hierarchies for accurate object detection and semantic segmentation

<https://www.jianshu.com/p/7dbdfe0aab2b>

深度学习之前一般使用传统的机器学习方法对图片进行特征工程以选取有效的特征，进而其于所选择的特征进行建模（如逻辑回归、SVM、贝叶斯网络等）以来识别出图片之上所具有的位置及其位置。

R-CNN使用CNN方法来提取图片中的特征以进行图片中物体检测。与之前的方法相比，它可以使用一个可共享参数（多个不同的类别之间）的CNN来得到图片的特征进行再基于此特征对每个类分别训练不同的SVM模型进行后续分类。



**输入：**待检测物体的图片

**输出：**图片中每个具体的物体名称以及它们在图片中所处的位置

**流程：**

1. 对每张图片，先采用传统的特征区域提取办法（此处使用Selective Search方法，其它可选的方法还有objectness，constrained parametric min-cuts (CPMC) 等）提取出固定数目的区域提案来（此处为2000个），然后将每个区域提案（Region proposal）进行变形为固定大小（此处为227x227）的图以作为接下来CNN网络的输入；
2. 每个由上步产生并变形为固定大小的图经过CNN网络后输出为一个特征向量（经过了若干CNN层后，由FC层变化为最终的一维特征向量）；
3. 由上步CNN处理过后的一维特征向量作为接下来SVM的输入进行SVM物体识别分类，从而决定出此区域方案属于哪一类；
4. 由CNN生成的特征向量来生成逻辑回归模型以对每个区域提案在图片中的位置进行调整。而此步骤没能反映在图中。

**模型训练：**

1. 先使用ILSVRC2012分类数据集来预训练CNN网络，此步骤中只使用分类的数据集使得我们的CNN网络学会基本的图像识别；
2. 使用要进行物体检测的数据集（此处为VOC0712数据集）所生成的有着固定大小的区域提案对trained过的CNN网络继续进行训练（此方法又叫finetune），最终得到了trained好的我们最终将采用的CNN网络；
3. 使用上个步骤trained好的CNN网络，我们可以将每个图片上的区域提案转换为一个定长的（此处为4096）特征向量；将此特征向量作为输入，我们按下来再训练N个（此处N为图片上物体可能的类别数目）SVM二分类模型来解决特征区域的分类问题，如此就能确定出我们最开始使用Selective Search方法所得到的图片上的某个区域提案有多大的概率属于哪一类物体；
4. 作者发现此方法检测出的物体的位置存在着较大的偏差。于是使用CNN网络中Pool5层输出的feature map作为特征，来重新训练了一个逻辑回归模型以来对特征提案的位置信息进行较正，最终得到了较高的MAP值。

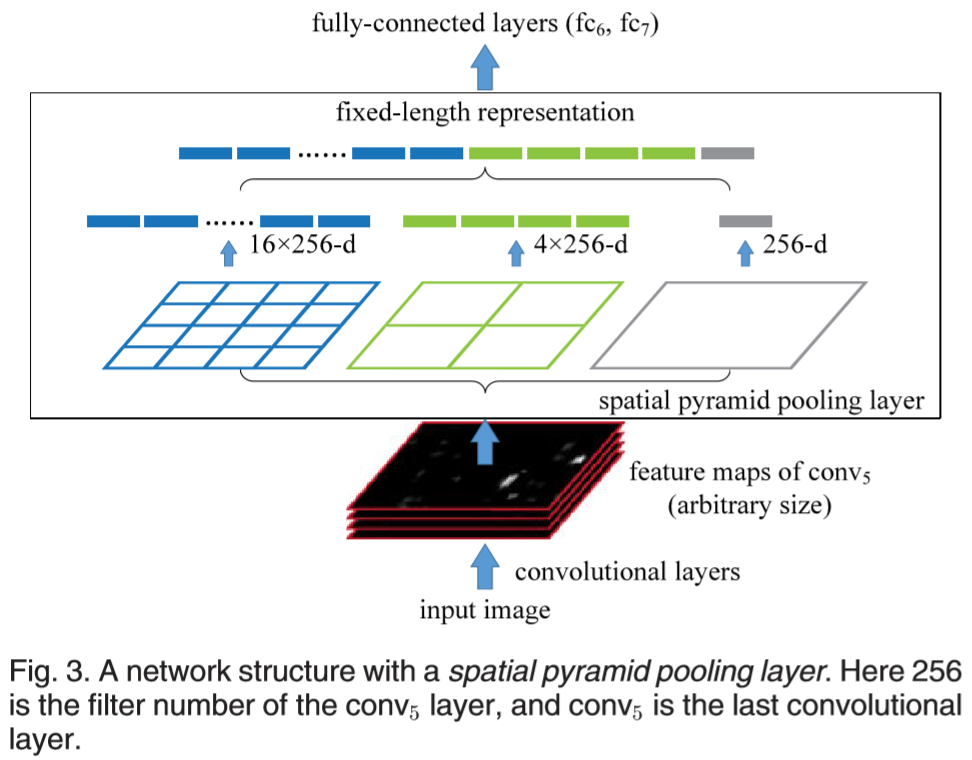
## 2015 SPP-Net

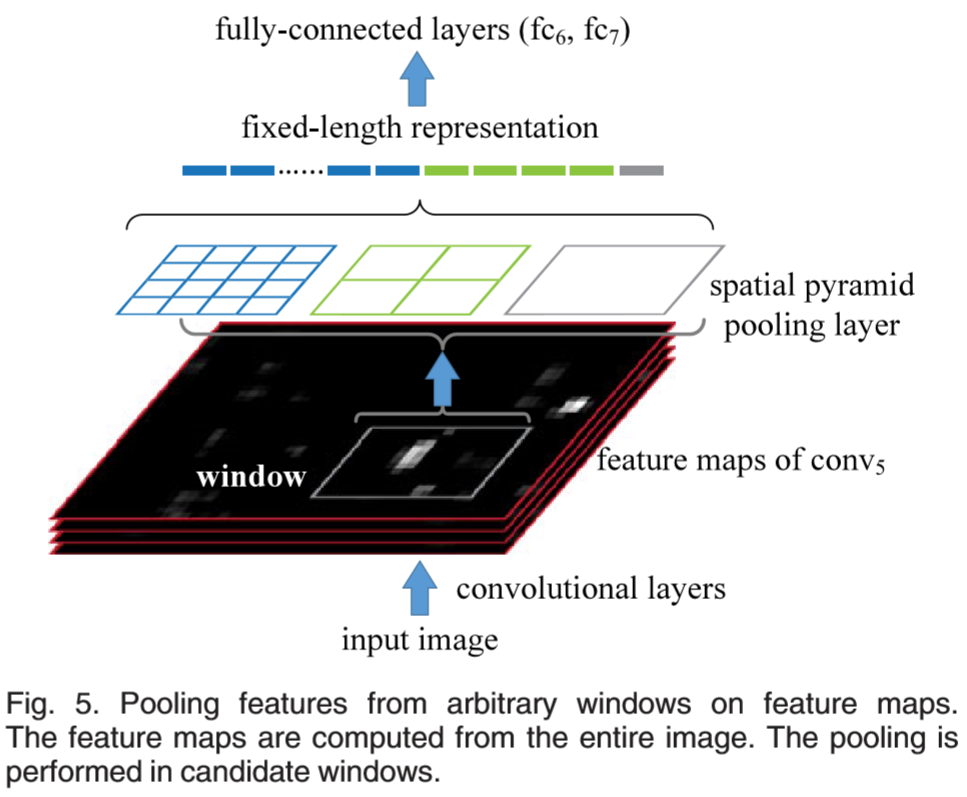
PAMI - Spatial pyramid pooling in deep convolutional networks for visual recognition

<https://www.jianshu.com/p/49ac77192e72>

物体分类或检测的CNN网络多是由两部分来组成，前端的CNN层用于特征提取，后端的FC层用于具体的问题处理如分类或物体位置识别。因为FC层的存在，意味着它只能接受固定大小的输入，即每次输入tensor的维数必须相同。这带来一个问题即我们只能使用统一大小的图片作为网络输入。我们不得不对图片进行一些处理如剪切（crop）或失真式变行（Warp），这些处理或会导致图片分辨率有变化，甚至可能使得输入图像有内容损失。

该文章一种可对任意大小的feature map输入进行Pooling处理来生成固定大小的空间金字塔式层化方法（Spatial pyramid pooling，简称SPP）。此种SPP层化方法被加在前端CNN层网络的最后一级feature map输出后可生成固定大小的特征来。





**优点：**

1. SPP层的使用可以让我们使用整幅图片作用前端CNN网络的输入，在最后的CNN层输出中，我们再提取出不同ROI窗口所对应的特征向量。这样以来，因为平均一幅图上面约有2000个ROI窗口，因此向对于普通R-CNN检测框架，我们能够省2000倍的计算量；
2. 为了增加模型检测的精度，作者有使用不同scale的输入图片来提取ROI的特征，这样使用一种类似于特征的组合的模式提高接下来的SVM分类模型的准确率。

## 2015 Fast\_R-CNN

ICCV - Fast\_R-CNN

<https://www.jianshu.com/p/b36f79c75369>

**R-CNN的缺点：**

1. 训练是多阶段的：获得图片的区域提案后，先使用这些提案训练CNN网络；然后去掉此CNN网络最后的Softmax层/FC层，使用CNN层输出的feature maps作为SVM分类器的输入，对SVM分类器进行训练；最后利用处理过的CNN层的feature maps作为输入训练一个逻辑回归模型以来对区域位置进行较正；
2. 训练时的空间与时间复杂度高：训练SVM与逻辑回归模型需要将图片区域提案通过CNN网络生成相应的特征向量；这些特征向量存储在硬盘当中；对于VGG16，一般需要2.5天的GPU时间来处理5000张VOC07图片，产生的特征向量需要数百GB的硬盘空间；
3. 检测慢：R-CNN网络需要对每个图片上的每一个区域提案进行三阶段处理，因此需要的时间极长，当使用VGG16网络时一般需要47秒来处理一副图片。

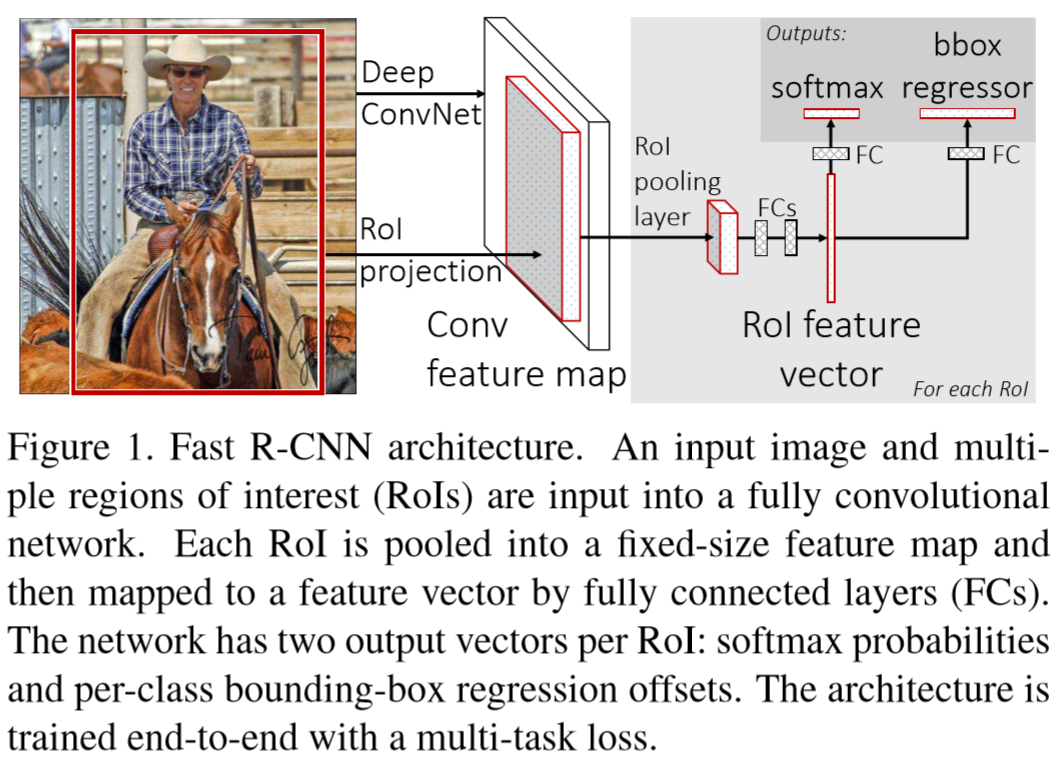
**SPP-Net的缺点：**

1. 训练是多阶段的；
2. 使用finetue方式进行训练模型增强时只对SPP层之后的FC层进行finetune，因此整体模型的准确度会受些影响。

**Fast\_R-CNN的改进：**

1. 取得了比R-CNN与SPP-Net更高的mAP准确率；
2. 使用单个阶段完成目标区域检测；（可以说是最本质的创新）
3. 单阶段训练，训练时对所有的层进行同步更新；
4. 因为是单阶段训练所以不需要额外的硬盘空间来存储中间特征。

下图为Fast\_R-CNN的基本网络结构，它最本质的创新在于直接将最后FC层输出的特征向量分别使用Softmax层与Regressor层来直接对区域方案的类别与位置进行预测与调整，这一网络结构后来也被其它的模型像Yolo与SSD采用。



**ROI池化层：**

ROI层的引入是Fast\_R-CNN对SPP-Net的最大借鉴；ROI池化层有两个参数H、W，分别表示池化层处理后最终能得到的feature map的高与宽；任意一个ROI特征（经过之间的CNN网络处理后）都可以视为一个（r,c,h,w）的四元组，其中（r,c）表示区域左上角的位置，（h,w）则为此区域的高宽大小；ROI层在对这么一个ROI特征四元组进行处理时，会先将其分为H x W个网格单元，每个单元的大小为h/H x w/W，然后分别对每个单元做MaxPool处理以得到一个极大值，这样最终就能得到一个大小为H x W的特征输出。本质上ROI层是只有一个空间池化级别的金字塔式空间池化层。

**前端CNN网络的初始化：**

对用于分类的CNN网络需要进行一些变形以来满足Fast\_R-CNN需求。首先需要将最后一个MaxPool层替换为ROI层以来输出与接下来FC层输入大小相匹配的feature maps；另外则需要将CNN分类网络最后端的FC层替换为两个兄弟层，一个为用于分类的Softmax层，另一个则用于区域位置检测的回归层，一般为L1回归；最后整个CNN网络的输入需要由两部分组成，分别为输入图片数据与对应的ROI区域位置提案。

**Fast\_R-CNN的训练损失函数：**

共有两个部分构成，分别对应两个层的输出（Softmax层与L1回归层）：ROI计算得到的相对于正确类别U的交叉熵损失； ROI最终调整过后得到的目标区域与正确目标区域之间的L1损失。

**模型使用：**

1. 以一张图片及其上的R个目标提案作为输入；
2. 前向运算得出每个ROI所对应的类别分布，与相应的目标位置框相对于输入的RoI窗口位置的偏移；
3. 使用与R-CNN类似的NMS（非最大值抑制）减少预测得到的目标检测框的数目

**启发：**

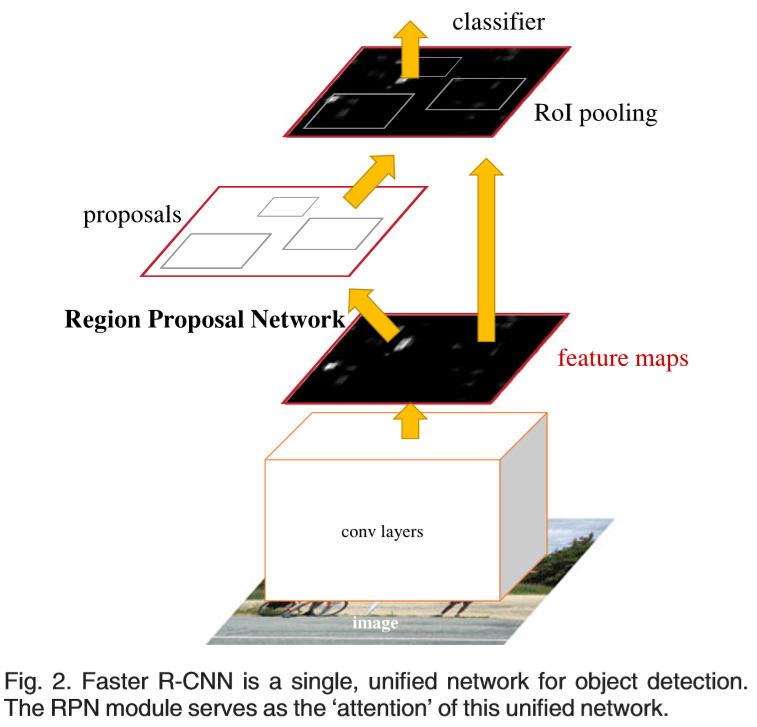
1. 通过多目标学习一起学习目标类别与位置，相对于像之前R-CNN或SPP-Net那样对目标类别与位置分别进行学习相比，可取得更好的mAP值；
2. 较深的网络在进行图片特征学习时会自动地学会图片大小无关的特征提取方法；因此在使用像VGG16这样较深的网络时，只使用单一缩放图片数据集进行模型训练在时间与精度权衡上效果更好。

## 2015 Faster\_R-CNN

NIP - Faster r-cnn: Towards real-time object detection with region proposal networks

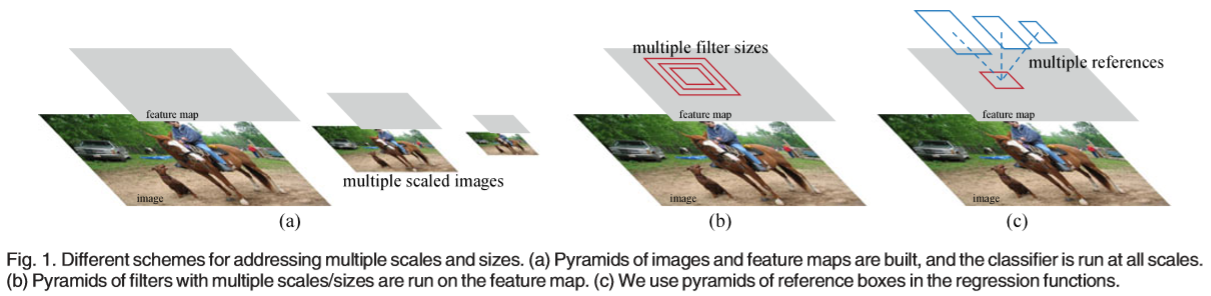
<https://www.jianshu.com/p/ede35536d62c>

提出RPN（Region Proposal Network）解决了Fast\_R-CNN对像Selective Search等传统意义上的特征区域提案（Region Proposal）方法的依赖，从而进一步使得模型检测速度与准度都有了较大的提高。可以简单认为Faster\_R-CNN = RPN + Fast\_R-CNN。RPN相对于像Selective Search等传统区域提案算法可以与Fast\_R-CNN主检测网络共享基础网络特征，因此能够大大地提高区域提案计算效率。

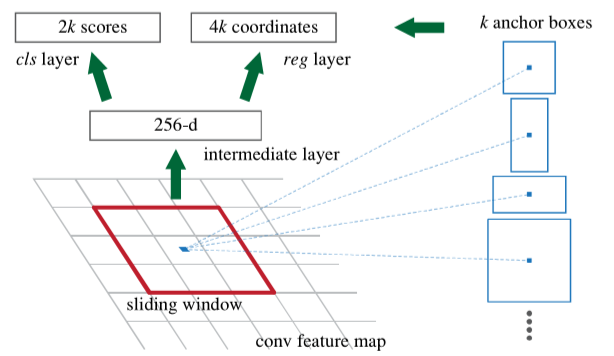


**RPN Anchor box：**

在R-CNN系列目标检测网络中，提案对其最终位置检测的准确率有极大的影响。很多CNN目标检测方法都努力提供具有不同scale / size大小的区域提案以最大范围地检测到任意scale / size的目标及其位置。通常有两种方法。一种是使用不同大小的图片作为CNN检测网络的输入（通过Resize获得不同大小的图片，同样一张图片需要resize成不同的size后喂入多次，后续size的输入图片则是对前面已经训练出的模型参数进行finetune）；另一种则是在使用CNN进行特征提取时选用不同大小的filter size（比如经典的Network In network中所提出的概念或者典型的Inception系列分类网络）。而在本文中作者提出了使用Anchor box来对输入feature map之中的任一点学习出不同大小/分辨率检测框的方法。以下图中可以看出三种方法的对比与区别。



对CNN特征提取网络最后生成的feature map上的每一点使用了9个相应的anchor box（分别具有不同的大小与分辨率）。通过Reg层及cls层就得到了9个anchor box分别对应的predict box的坐标（4\*k）及其是否包含有目标的概率大小（k个2分类）。



**RPN损失函数：**

与Fast\_R-CNN中介绍过的loss函数类似，RPN网络的loss函数也包含分类loss与位置回归loss两个部分。

**RPN训练：**

对RPN的训练跟之前Fast\_R-CNN的训练类似，同样是将前端CNN特征提取网络使用Imagenet 数据预训练过的权重进行初始化，然后再使用检测数据集（如VOC等）进行finetune。

RPN网络输出中数千个区域提案，大多数为负样本。负样本数目远大于正样本，需要对负样本进行采样。是机器学习中保持数据样本分布平衡的典型策略。

**Faster\_R-CNN的训练：**

1. 交叉式训练：首先训练出RPN网络来；然后使用它生成的图片区域提案再训练Fast\_R-CNN目标检测网络；接下来再使用此步骤生成的权重初始化整个Faster\_R-CNN网络来finetune RPN的部分；最终再使用上步骤生成的权重初始化整个Faster\_R-CNN，然后固定RPN网络部分的权重不变化，finetune Fast\_R-CNN部分ROI之后的层，最终得到训练好的Faster\_R-CNN模型。该方案实验效果最好。
2. 近似联合训练：同时训练RPN与Fast\_R-CNN两个网络部分；相当于它有四个loss（RPN与Fast\_R-CNN各两个），然后基于四个loss分别向后做backward。然后在前端CNN网络的最后一层backward时，它会对分别来自于RPN与Fast\_R-CNN的参数梯度进行加总以作为它输出的梯度向后进行backward计算。在这整个过程中，我们忽略了由Fast\_R-CNN对RPN所输出的区域提案的backward，所以称此方法为近似联合训练。作者在文章中指出此方法能够得到与上述交叉式训练方法近似的精度，同时还能减少约25%-50%的模型训练时间。
3. 联合训练：与上述近似联合训练方法不同，它考虑了Fast\_R-CNN对RPN所输出区域提案的backward计算，这样更能反映偏微分编程模型本质。但因为自ROI pool之后对区域提案的backward的实现比较复杂，因此作者并未对此过多讨论。

相对于之前SS + Fast\_R-CNN的state of art方法，Faster\_R-CNN能够在精度与速度上都有较大提高。显然其最大的贡献在于RPN的提出与使用。

## 2016 R-FCN

NIP - R-fcn: Object detection via region-based fully convolutional networks

<https://www.jianshu.com/p/bb326a16658d>

<https://www.jianshu.com/p/30b57b36add5>

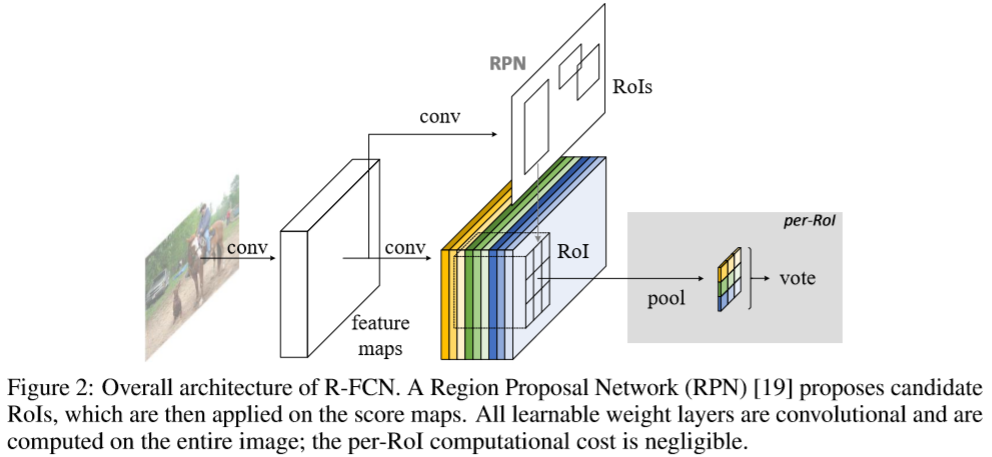
事实上，一些state of art的图片分类网络均为全卷积网络，如ResNet、GoogLeNet等。所以很自然地，是否可以将目标检测的网络也用全卷积网络来实现？事实证明，如果简单地丢弃全连接层（实现融合特征和特征映射），检测的效果会很差。作者认为这主要来源于这样的一对矛盾：

1. 图像分类：要求图像具有平移不变性（translation invariance）
2. 目标检测：要求图像具有位置敏感性（translation variance）

和Faster R-CNN一样，R-FCN也是 基于region proposal的两级检测架构。

对于region-based的检测方法，以Faster R-CNN为例，实际上是分成了几个subnetwork，第一个用来在整张图上做比较耗时的conv，这些操作与region无关，是计算共享的。第二个subnetwork是用来产生候选的boundingbox（如RPN），第三个subnetwork用来分类或进一步对box进行regression（如Fast RCNN），这个subnetwork和region是有关系的，必须每个region单独跑网络，衔接在这个subnetwork和前两个subnetwork中间的就是ROI pooling。

我们希望耗时的卷积都尽量移到前面共享的subnetwork上。因此，和Faster RCNN中用的ResNet（前91层共享，插入ROI pooling，后10层不共享）策略不同，本文把所有的101层都放在了前面共享的subnetwork。最后用来prediction的卷积只有1层，大大减少了计算量。



Faster R-CNN中，ROI Pooling层直接对ROI进行分块池化输出用于分类和回归的特征向量。

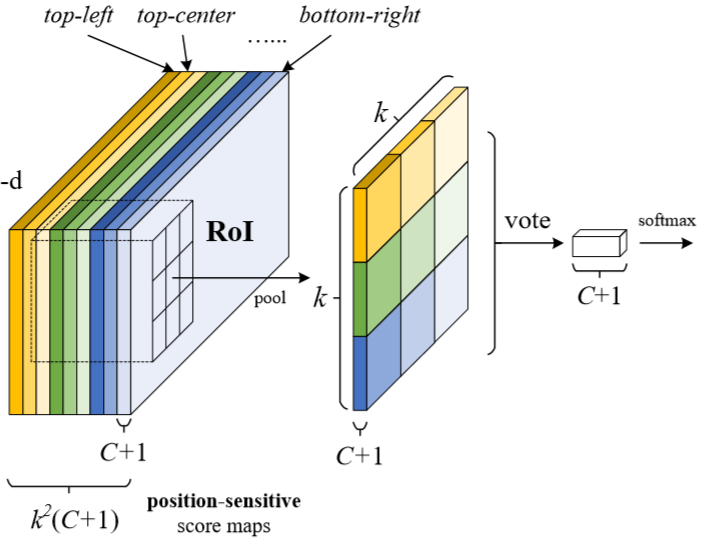
R-FCN中，则将每一个ROI划分成k×k个格，池化输出每个格的位置得分，再通过投票方式得到 ROI 最后的输出特征向量。的首先生成 k^2(C+1) 通道大小的输出。其中，C 为类别数（+1为背景），k^2 表示将ROI区域划分成 k×k个格，如上图所示。如 k=3，则对应9个格

**Backbone网络：**

ResNet101。去除原始网络最后的平均池化层和FC层，保留100层的卷积层用于特征提取。为了降维，100层卷积层之后又添加了一层1×1×1024的卷积层，使输出维度变成1024（原始的是2048）。之后再接一层卷积层用于产生得分图。

**位置敏感得分图&位置敏感ROI Pooling：**

将w×h大小的ROI划分成k×k个格。对每个格进行位置敏感池化操作。k^2个得分图通过投票（eg.求均值）后得到ROI上 C+1 维的输出向量。后接一个softmax层得到每一类的最终得分，并在训练时计算损失，如下图。



**ROI的位置回归：**

R-FCN的位置回归方式与R-CNN和Fast R-CNN相似。上面说到，在base网络的 feature map 之后连接了一层卷积用于产生k^2(C+1)维的位置敏感得分图。此处仍然从base网络上连接一个 4k^2 通道的得分图（与位置敏感得分图并列），用来做 ROI 的坐标微调。同样对这个 4k^2 大小的得分图进行 ROI Pooling操作，输出 t=(t\_x, t\_y, t\_w, t\_h) 大小的 4 维坐标向量，用于位置回归。

**训练的相关细节：**

损失函数与Fast RCNN相似，分类部分为交叉熵损失，回归部分为平滑L1损失，总的损失为这两部分的和。采用类似Faster R-CNN中的4步训练策略交替训练RPN网络和R-FCN网络。

## 2017 Mask\_R-CNN

ICCV - Mask r-cnn

<https://www.jianshu.com/p/0db565b2ef7d>

Mask RCNN是在Faster-RCNN的基础上改进后被用于解决图像instance segmentation的问题。相对于原来的Faster\_RCNN主干框架，它在网络的头上引入了另外一条FCN并行分支用来检测ROI的mask map信息。这样最终它的头部共有三条并行的分支分别用来处理ROI区域的类别识别、目标框位置回归及相应的mask map回归。

# Yolo

## 2016 Yolo\_v1

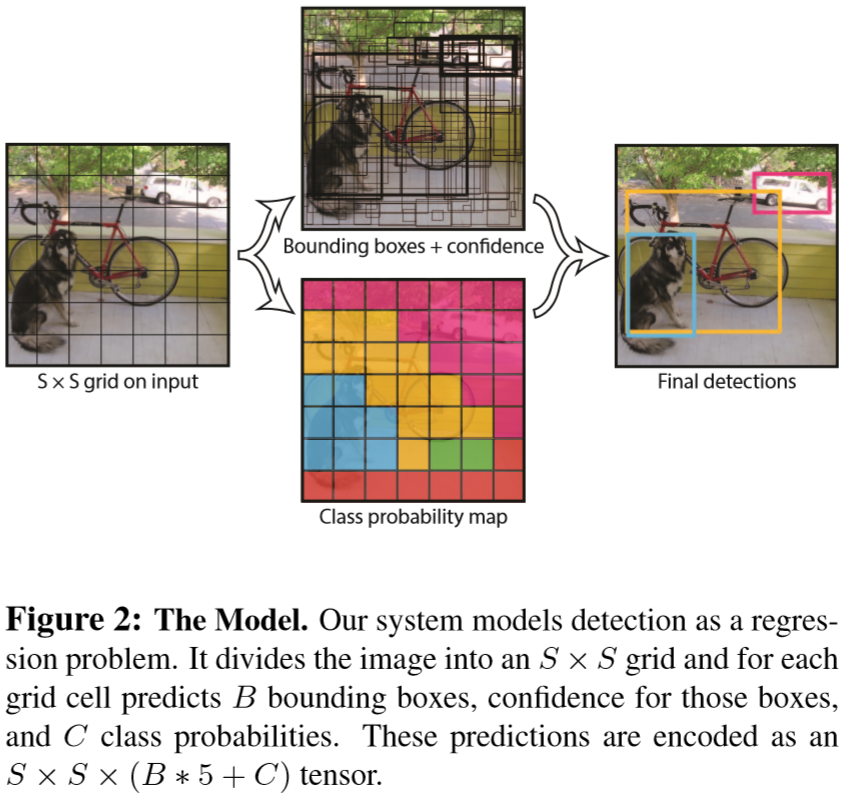
CVPR - You Only Look Once Unified, Real-Time Object Detection

<https://www.jianshu.com/p/32c77fe3f5cd>

<https://blog.csdn.net/guleileo/article/details/80581858>

主流的CNN目标检测框架可分为两类：包含Region proposals提取阶段的两阶段目标检测框架如R-CNN系列像R-CNN/Fast-RCNN/Faster-FCNN/R-FCN等；端到端的单阶段目标检测框架如Yolo系列像Yolo-v1/Yolo-v2/Yolo-v3及SSD，和最近Facebook提出的RetinaNet等。两阶段的检测框架相对准确率（mAP值）更高，但其速度较慢，很难满足图片或视频实时性处理要求；单阶段的检测框架相对准确率较低，但却能在保证一定准确率的情况下，拥有更快、甚至实时的推理速度

YOLO的核心思想就是利用整张图作为网络的输入，直接在输出层回归boundingbox（边界框）的位置及其所属的类别。



将一幅图像分成SxS个网格（grid cell），如果某个object的中心落在这个网格中，则这个网格就负责预测这个object。

每个网格要预测B个bounding box，每个bounding box除了要回归自身的位置(x, y, w, h)之外，还要附带预测一个confidence值。这个confidence代表了所预测的box中含有object的置信度和这个box预测的有多准这两重信息。

每个网格还要预测一个类别信息，记为C类。则SxS个网格，每个网格要预测B个boundingbox还要预测C个categories。输出就是SxSx(5\*B+C)的一个tensor。

class信息是针对每个网格的，confidence信息是针对每个bounding box的。

每个网格预测的class信息和bounding box预测的confidence信息相乘，就得到每个bounding box的class-specific confidence score。得到每个box的class-specific confidence score以后，设置阈值，滤掉得分低的boxes，对保留的boxes进行NMS处理，就得到最终的检测结果。

**损失函数：**

每个grid有30维，这30维中，8维是回归box的坐标，2维是box的confidence，还有20维是类别。其中坐标的x,y用对应网格的offset归一化到0-1之间，w,h用图像的width和height归一化到0-1之间。

设计损失函数，让这个三个方面得到很好的平衡。作者简单粗暴的全部采用了sum-squared error loss来做这件事。

**这种做法存在几个问题：**

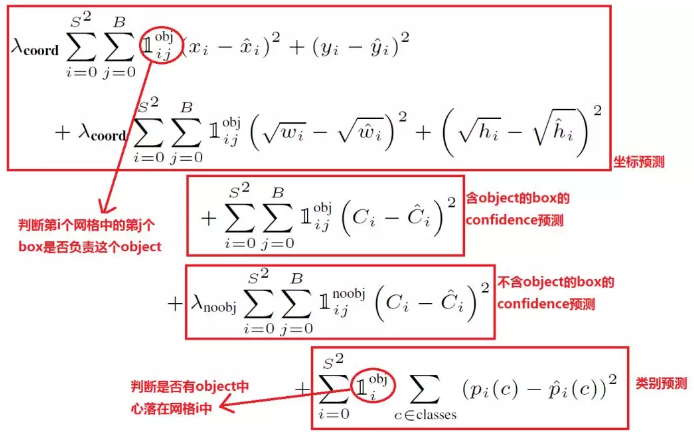
1. 8维的localizationerror和20维的classificationerror同等重要显然是不合理的；
2. 如果一个网格中没有object（一幅图中这种网格很多），那么就会将这些网格中的box的confidencepush到0，相比于较少的有object的网格，这种做法是overpowering的，这会导致网络不稳定甚至发散。

**解决办法：**

1. 更重视8维的坐标预测，给这些损失前面赋予更大的loss weight,记为在pascalVOC训练中取5。
2. 对没有object的box的confidence loss，赋予小的loss weight，记为在pascalVOC训练中取0.5。
3. 有object的box的confidence loss和类别的loss的loss weight正常取1。
4. 对不同大小的box预测中，相比于大box预测偏一点，小box预测偏一点肯定更不能被忍受的。而sum-square error loss中对同样的偏移loss是一样。作者用了一个比较取巧的办法，就是将box的width和height取平方根代替原本的height和width。这个参考下面的图很容易理解，小box的横轴值较小，发生偏移时，反应到y轴上相比大box要大。（也是个近似逼近方式）

一个网格预测多个box，希望的是每个box predictor专门负责预测某个object。具体做法就是看当前预测的box与groundtruth box中哪个IoU大，就负责哪个。这种做法称作box predictor的specialization。

**最终损失函数：**



这个损失函数中：

1. 只有当某个网格中有 object 的时候才对 classification error 进行惩罚。
2. 只有当某个box predictor 对某个ground truth box负责的时候，才会对box的coordinate error进行惩罚，而对哪个ground truth box负责就看其预测值和ground truth box的IoU是不是在那个cell的所有 box 中最大。

**缺陷：**

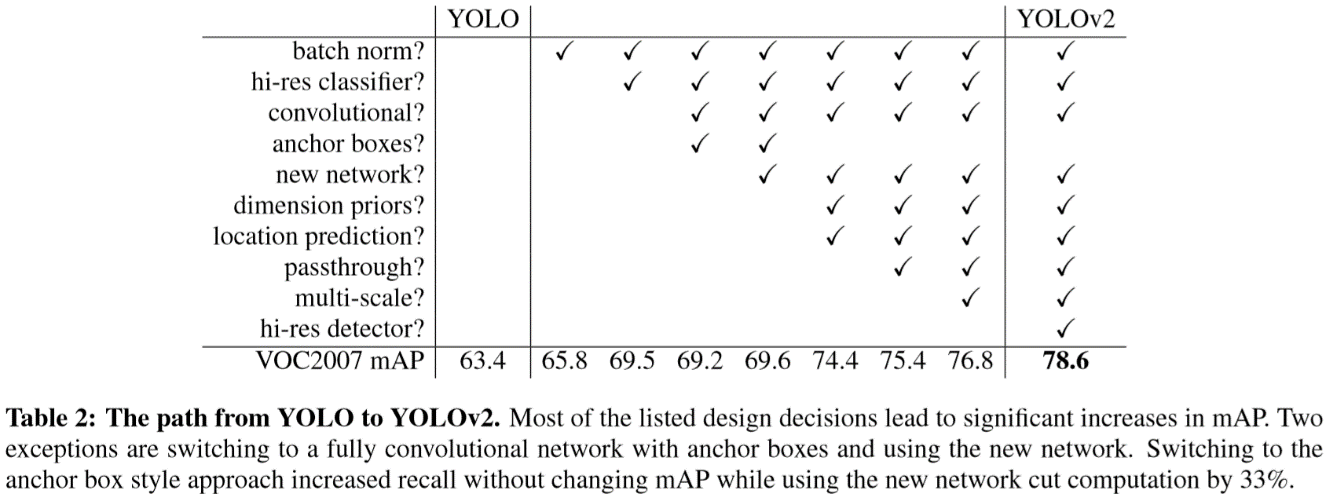
1. 由于输出层为全连接层，因此在检测时只支持与训练图像相同的输入分辨率
2. 虽然每个格子可以预测B个bounding box，但是最终只选择只选择IOU最高的bounding box作为物体检测输出，即每个格子最多只预测出一个物体。当物体占画面比例较小，如图像中包含畜群或鸟群时，每个格子包含多个物体，但却只能检测出其中一个。
3. 对于非常规的物体形状或比例，如同一类物体出现的新的不常见的长宽比和其他情况时，泛化能力偏弱。
4. 采用了多个下采样层，网络学到的物体特征并不精细，影响检测效果。
5. 损失函数中，大物体IOU误差和小物体IOU误差对网络训练中loss贡献值接近（虽然采用求平方根方式，但没有根本解决问题）。因此，对于小物体，小的IOU误差也会对网络优化过程造成很大的影响，从而降低了物体检测的定位准确性。

## 2017 Yolo\_v2

CVPR - YOLO9000 Better, Faster, Stronger

<https://www.jianshu.com/p/b02f64e0d44b>

**相对V1的改进：**



1. **BN（batch normalization）的引入**

使用 Batch Normalization 对网络进行优化，让网络提高了收敛性，同时还消除了对其他形式的正则化（regularization）的依赖

1. **finetune时高精度分类器的使用**

在Yolo v2中，在拿到224x224的分类数据集train过的模型后先使用448x448的分类数据集finetune上10个epochs，然后再使用448x448的目标检测数据集进行接下来的目标检测工作的finetune

1. **使用卷积操作的Anchor boxes**

相对于Yolo v1中直接使用FC来预测目标框的类别与位置，使用卷积生成anchor boxes位置与类别的方法会带来mAP约0.3的下降，但会导致较大的召回率（Recall ratio）提升，约7%。这使得这一模型可进一步改良、提升的空间加大。

1. **使用K-means cluster来选取anchor boxes**

直接在目标检测训练数据集上对所有的目标框进行k-means聚类最终得到了Avg IOU与模型复杂度均不错的anchor boxes组合。

1. **细粒度特征的使用**

SSD使用多尺度的feature maps特征。Yolo v2也吸取了此一优点。但它并不像SSD那样分别在不同的feature maps之上对不同尺度大小的anchor box进行预测，而是将拥有较细粒度特征的层变形后（使得与后面粗粒度的特征层有着一样的size，能在channel level上进行合并）与后面粗粒度的特征层结合在一起用于后续的anchor box预测。

1. **多尺度训练**

每训练10个epochs随机地从一个组合{320;352;…608}中选取（注意它们都是32的倍数，因为darknet网络的步长为32）一个数作为输入的image size。可有效地使得网络学会去自动识别各种尺度大小的图片。

1. **使用Darknet-19**

文章提出了一种新的训练方法–联合训练算法。这种算法可以把这两种的数据集混合到一起。使用一种分层的观点对物体进行分类，用巨量的分类数据集数据来扩充检测数据集，从而把两种不同的数据集混合起来。

联合训练算法的基本思路就是：同时在检测数据集和分类数据集上训练物体检测器（Object Detectors ），用监测数据集的数据学习物体的准确位置，用分类数据集的数据来增加分类的类别量、提升鲁棒性。

YOLO9000 就是使用联合训练算法训练出来的，他拥有 9000 类的分类信息，这些分类信息学习自ImageNet分类数据集，而物体位置检测则学习自 COCO 检测数据集。

YOLO 9000 的网络结构允许实时地检测超过9000种物体分类，这归功于它能同时优化检测与分类功能。使用 WordTree 来混合来自不同的资源的训练数据，并使用联合优化技术同时在 ImageNet 和 COCO 数据集上进行训练，YOLO9000 进一步缩小了监测数据集与识别数据集之间的大小代沟。

## 2018 Yolo\_v3

arXiv - YOLOv3: An Incremental Improvement

<https://www.jianshu.com/p/94e7c737e851>

Yolo v3是在之前Yolo v2的基础上引入了一些小的改进，使得最终的mAP值达到了与SSD近似的程度，同时其速度更是要快上3倍：

1. **Bounding box预测**

它在判断anchor box的正负时，每个ground truth box只会选用一个与它有最大IOU的anchor box作为正的anchor box，而不再使用一个threshold（一般为0.5），然后将超过此threshold值的anchor box也视为正样本框。这一点是它与SSD或Faster-RCNN系列模型不同的地方。

1. **类预测**

Yolo v3使用多目标的方式进行分类，而不是softmax。它在最终的loss层中使用逻辑回归来得到每个bounding所属的类别分布，然后再使用cross-entropy来计算training loss。

1. **multi-scales的采用**

类似于FPN网络，Yolo v3中采用了相似的思想使用后面三个Conv层的特征来进行预测。在这些层之间也像FPN那样有up-to-down和down-to-up的数据通路。此外它anchor box dimensions的选取同Yolo v2一样也是通过对训练数据集做k-means聚类得到的。它会将选择出的9个anchor box平均地分布到三个scales的feature maps当中。

1. **darknet-53**

Yolo v3中引入了更为强大的特征提取网络darknet-53。

**结论：**

1. 像RPN等网络那样预测anchor box的位置偏离等信息会造成模型训练时的震荡。
2. 直接线性地预测x,y的偏离而不是使用logistic action同样被证明是不可取的。
3. Focal loss的引入同样会使最终的mAP值下降几个点。分析认为在预测anchor box位置时，objectness的引入即其在最终loss计算时所起的作用表明潜在的过多的negative box不会对classification loss分支产生较大影响。
4. 双重IOU计算标准的引入造成模型精度不一表明当下multi-scales的引入使得Yolo v3可以较好地预测出小scale的目标但却在大scale的目标上产生了不少的错误。

# SSD

## 2016 SSD

ECCV - SSD Single Shot MultiBox Detector

<https://www.jianshu.com/p/f9c973c43bdc>

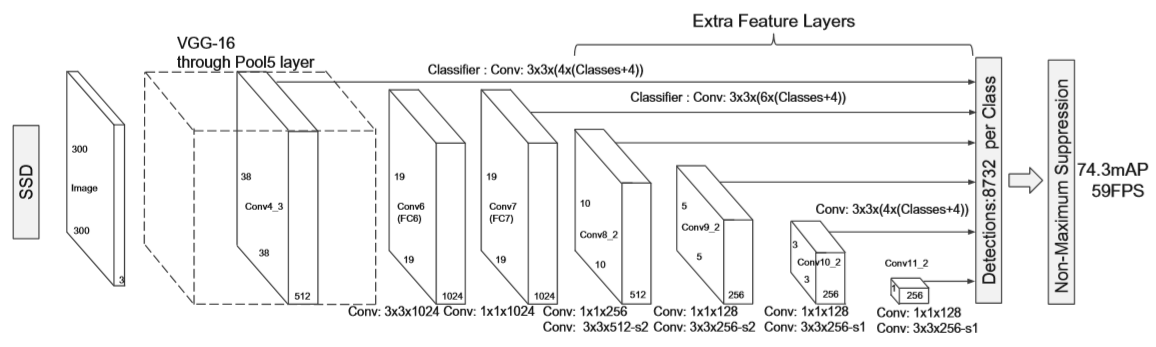
anchor，default box，prior box表示的是同一个意思。anchor技术的鼻祖是DeepMulti Box，这篇论文里首次提出使用prior(先验框)，并且提出使用prior做匹配。后面的众多工作，比如RCNN系列，YOLO系列都使用了anchor这个术语，而在SSD中anchor又叫default box，本质上是同一个东西。

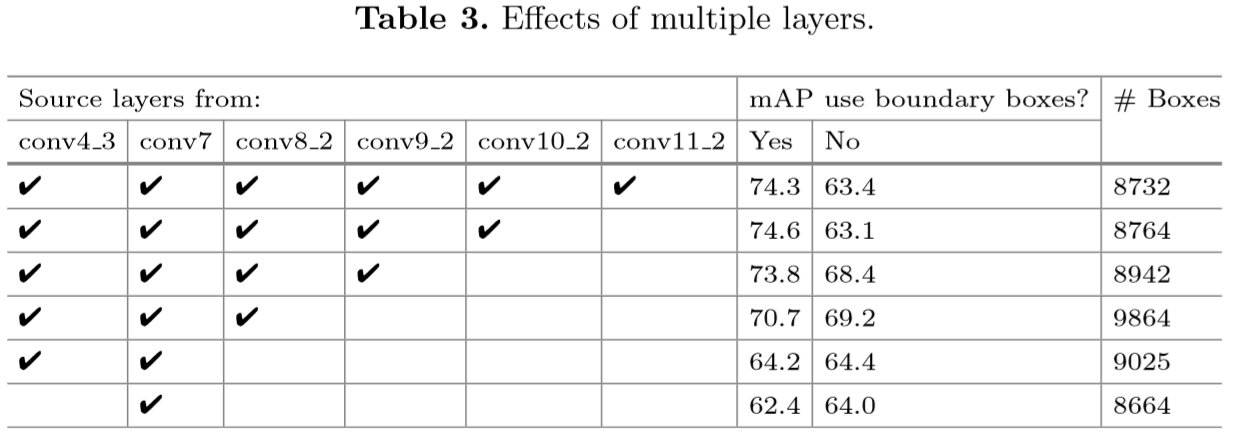
SSD效果好主要有三点原因：

1. 没有使用FC层
2. 多尺度
3. 设置了多种宽高比的anchor
4. 数据增强

**多尺度：**

SSD使用6个不同特征图检测不同尺度的目标。低层预测小目标，高层预测大目标。





**Anchor：**

anchor作用：通过anchor设置每一层实际响应的区域，使得某一层对特定大小的目标响应。

在特征图的每个像素点处，生成不同宽高比的anchor,论文中设置的宽高比为{1,2,3,1/2,1/3}。假设每个像素点有k个anchor，需要对每个anchor进行分类和回归，其中用于分类的卷积核个数为ck(c表示类别数)，回归的卷积核个数为4k。

**数据增强：**

SSD中使用了两种数据增强的方式

1. 放大操作：随机crop，patch与任意一个目标的IOU为0.1,0.3,0.5,0.7,0.9，每个patch的大小为原图大小的[0.1,1],宽高比在1/2到2之间。能够生成更多的尺度较大的目标
2. 缩小操作：首先创建16倍原图大小的画布，然后将原图放置其中，然后随机crop，能够生成更多尺度较小的目标

**缺点：**

1. SSD对小目标的检测效果一般，作者认为小目标在高层没有足够的信息。
2. 论文中提到的anchor设置没有对齐感受野，通常几个像素的中心位置偏移，对大目标来说IOU变化不会很大，但对小目标IOU变化剧烈，尤其感受野不够大的时候，anchor很可能偏移出感受野区域，影响性能。
3. 论文提到根据特定数据集设计default box，在YOLOV2中使用聚类的方式初始化anchor，能够更好的匹配到ground truth,帮助网络更好的训练

## 2017 DSOD

ICCV - Dsod: Learning deeply supervised object detectors from scratch

提出了不需要预训练模型的检测算法，可以看作SSD + DenseNet。以前的目标检测算法基本都是先在 ImageNet 数据库上进行预训练，然后再微调。这个微调也叫做迁移学习 transfer learning。

**微调的优点：**

1. 有很多好的模型是开源的，可以直接将他们用于目标检测；
2. 微调可以快速得的奥最终的模型，需要的训练数据也相对较少。

**微调的缺点：**

1. Limited structure design space，基于ImageNet 分类数据库的预训练模型大，参数太多。直接将它们用于目标检测，导致模型结构灵活性差，难以改变网络结构，计算量也大，限制其应用场合。
2. Learning bias，因为分类和检测任务的损失函数和 类别分布是不一样的，我们认为两者的搜索/优化空间是不一样的，所以在分类上预训练可能会导致在检测上的得到局部最优，而不是全局最优解
3. Domain mismatch，尽管微调可以减少不同目标类别分布的差异性，但是对于一些目标域为 深度图、医学图像和ImageNet的差异性太大, 微调效果不理想

**文章贡献：**

1. 从零开始训练一个好的目标检测网络
2. 设计这样的网络有什么原则可以遵守

**设计原则：**

1. Principle 1: Proposal-free

基于候选区域提取的方法无法从零开始训练，不收敛。只有proposal-free 方法可以从零开始训练并收敛。猜测原因是RoI pooling需要好的初始化输入，作者选择了基于SSD来改进

1. Principle 2: Deep Supervision

主要是参考 DenseNet 中 更多的 skip connections 实现 supervised signals 传递

1. Principle 3: Stem Block

受Inception-v3 [33]and v4 [31] 的启发 我们定义 Stem Block 为三个 3×3卷积层和一个 2×2 最大池化层，实验发现这么设计可以提升性能

1. Principle 4: Dense Prediction Structure

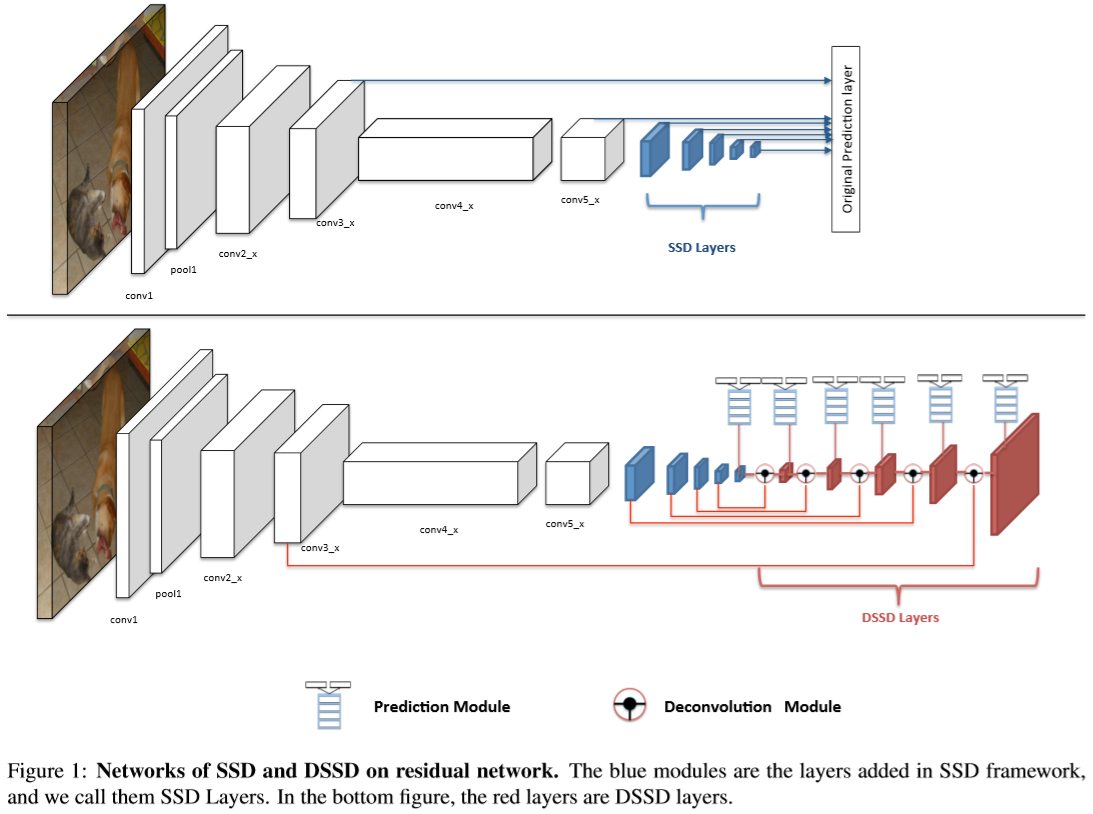
和SSD类似，DOSD也是在6个尺度特征图进行目标检测，再融合，和SSD 有所区别的是怎么利用特征图的，DOSD是 Learning Half and Reusing Half。

## 2017 DSSD

arXiv - DSSD Deconvolutional Single Shot Detector

<https://www.jianshu.com/p/3522f7f76ac6>

DSSD相当原来的SSD模型主要作了两大更新。一是替换掉VGG，而改用了Resnet-101作为特征提取网络，并在对不同尺度feature maps特征进行default boxes检测时使用了更新的检测单元；二则在网络的后端使用了多个deconvolution layers以有效地扩展低维度信息的contextual information，从而有效地提高了小尺度目标的检测。

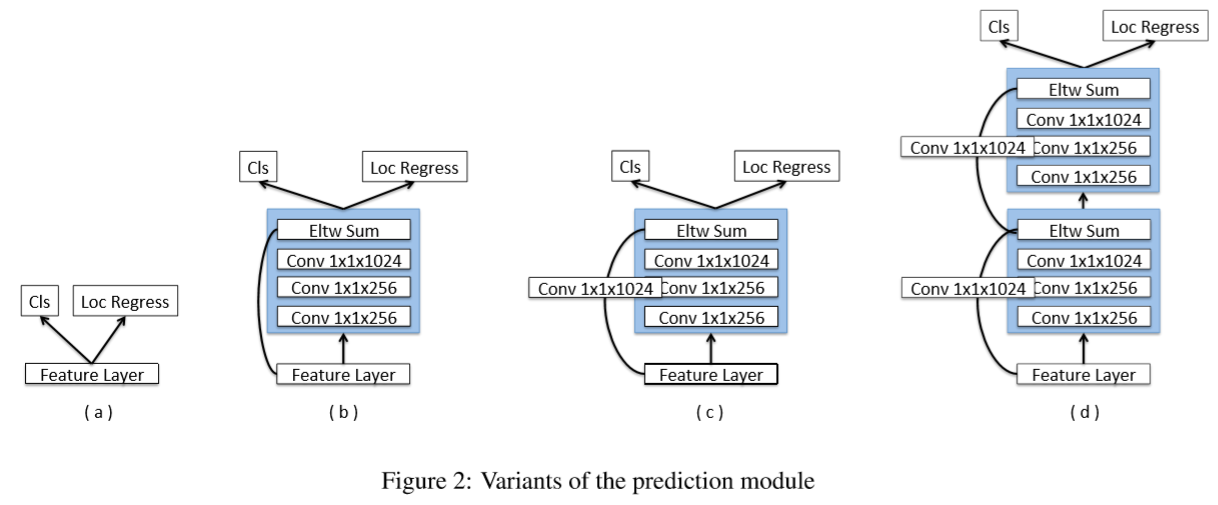


1. **使用Resnet-101而非VGG**

用更强大的Resnet-101来代替之前在SSD中使用的VGG作为模型的特征提取网络。在Resnet-101模型后端添加了一些Residual模块以来加强原来模型的特征表达

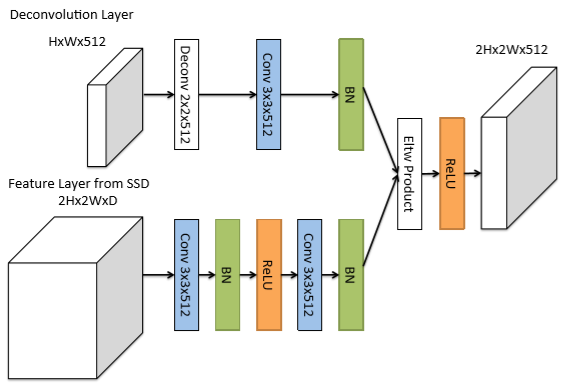
1. **新的预测模块**

在原来SSD模型中，采用了直接使用3x3的conv来在feature maps组合上进行预测。作者对每个需要预测的feature maps先使用residual模块进行处理，然后再进行box分值预测。（最终采用了下图中的C）



1. **反卷积模块**

DSSD在模型后端添加了多个反卷积模块来扩大模型在小尺度上的high level特征信息。而这种反卷积输出的feature maps又会与模型前端卷积层的相同尺度（feature map size）的feature maps进行元素级的乘法（element-wise product）来生成相应尺度的feature maps。下图中可看到某一尺度上反卷积模块的具体实现及其与相同size的卷积模块之间的结合。那么为何使用deconvolution来扩展feature maps size而非直接使用双线性插值，以及为何使用元素级乘而非加？答案是作者实验试出来的，后者的结果比前者更好。



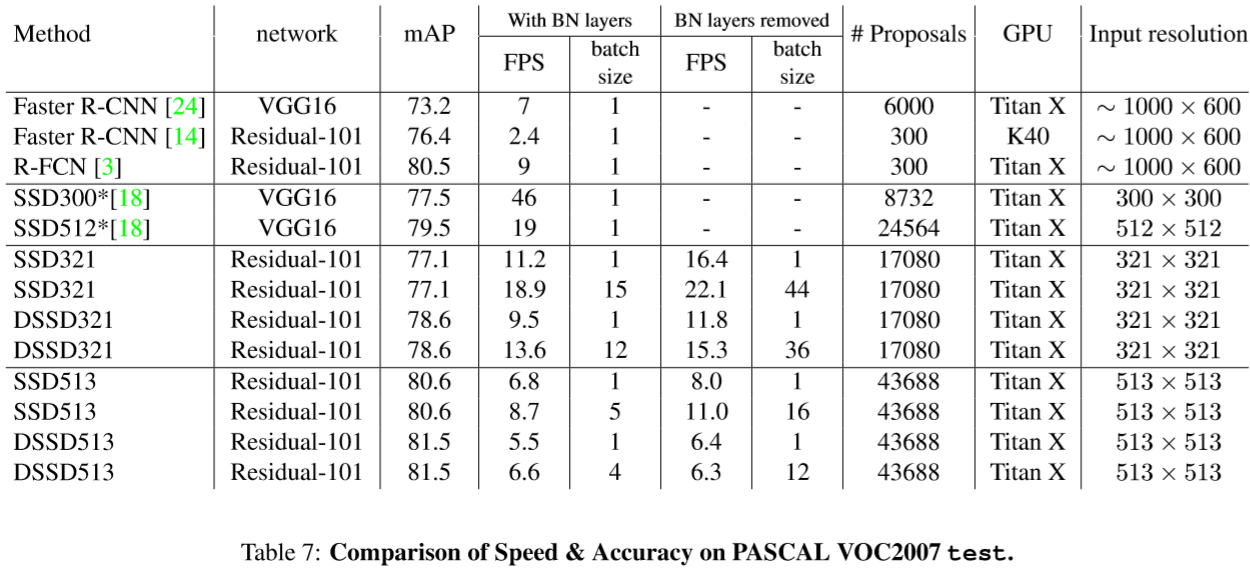
1. **模型训练**

在选取default boxes的scale时，它借鉴了Yolo v2中的做法，即对training dataset中已有的ground truth boxes进行聚类分析。然后基于此aspect ratio分布，在原来的基础上增添了一种default box。具体训练时，选使用与SSD相似的策略训练出SSD部分的网络，然后再固定下来SSD部分的网络权重，finetune训练deconvolution层的权重。

1. **模型推理**

在进行推理时，作者对BN层作了个小的优化，即将BN中所用的scale / shift等计算转换到conv中来进行（即对conv中使用的w与b参数进行预处理）。数学上等价。这样就省掉了bn层的计算，节省计算时间与内存需求。

**实验结果：**



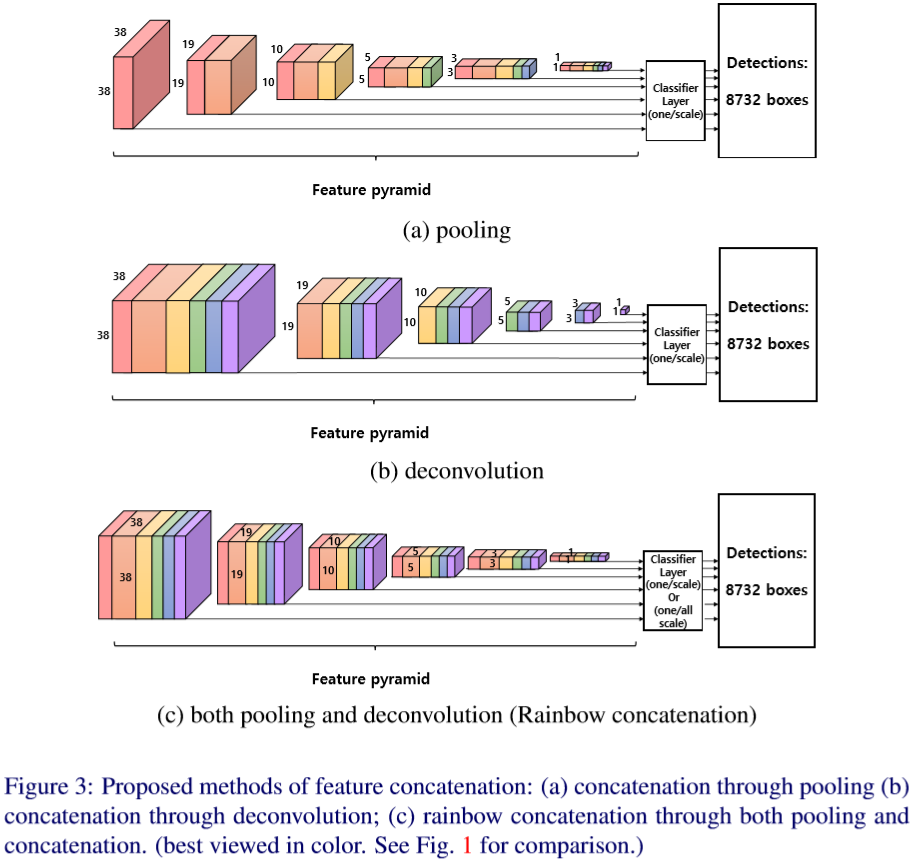
## 2017 RSSD

arXiv - Enhancement of SSD by concatenating feature maps for object detection

<https://zhuanlan.zhihu.com/p/47676544>

RSSD解决SSD中存在的两点不足：

1. 同一个目标被特征金字塔上的多个检测分支，检出多个bbox；
2. 小尺度目标的漏检；



RSSD在特征金字塔的不同检测分支上，尝试三种方案以提升feature map的通道数，使得classifier network能更好地利用不同尺度feature map层之间的关联性；

1. (a)：低层、高分辨率feature map做pooling后，与高层、低分辨率feature map做concate；

优势：高层检测分支上，不仅拥有大感受野，也能获得低层feature map上传递而来的目标细节信息，加上自身的高语义信息，组合的特征输入classifier network效果更好

劣势：信息流的传递是单向的，classifier network获得的信息依旧不够全面；

1. (b)：高层、低分辨率feature map做de-conv / upsample后，与低层、高分辨率feature map做concate；

优势：有点类似FPN的组织方式；

劣势：信息流的传递是单向的

1. (c)：RSSD采用的方式（rainbow concate），结合pooling + de-conv，concate所有高、低层feature map上的特征；

优势：信息流的传递是双向的，高低层feature map上的信息做到了融会贯通；各层feature map通道数统一为2816（512+1024+512+256+256+256），因此各个分支后接的classifier network之间可以同享同一套权重参数；

**RSSD与SSD对比：**

RSSD：RSSD中各个classifier network间参数共享，需要训练的参数就更少，训练速度就更快；且各个尺度的目标都可以用于更新classifier network的同一套参数，不像SSD中，特定尺度的目标仅用于更新检测该尺度目标的分支上的参数；这样classifier network可以获得更多数量、更多尺度的目标的监督信息用于训练，泛化能力更强，且利于数据量不足时的模型训练；

SSD：如果训练图像中，刚好没有用于检测某个特定目标的尺度，那么该尺度对应的检测分支在这轮训练中，将学不到任何信息；但RSSD因为将classifier network统一成了权重参数共享的方式，即使训练图像上的特定尺度目标，不在当前检测分支的处理范围内，但该目标肯定会被其他检测分支上所捕获，更新的模型参数也同样有利于当前检测分支的目标检测。

# Other

## 2013 OverFeat

arXiv - Overfeat: Integrated recognition, localization and detection using convolutional networks

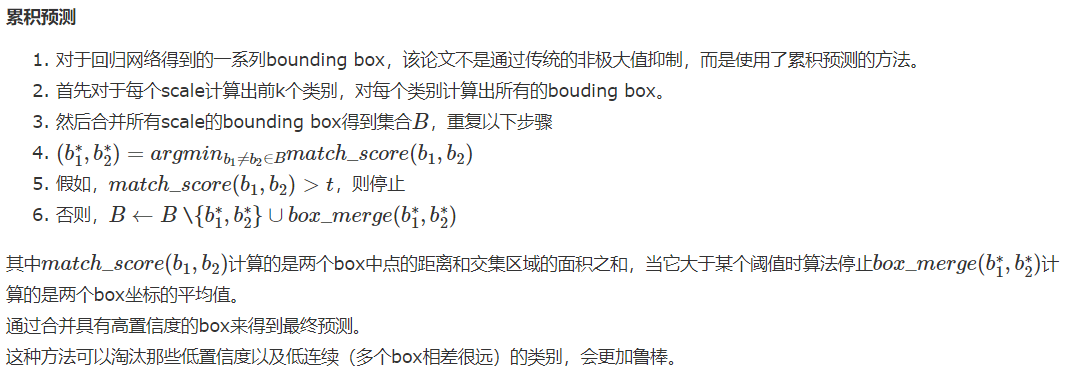
1. 用一个共享的CNN来同时处理图像分类，定位，检测三个任务。
2. 用CNN有效地实现了一个多尺度的，滑动窗口的方法，来处理任务。
3. 通过累积预测来求bounding boxes（而不是传统的NMS）。

**分类：**基于alexNet，使用全卷积和offset池化。

**池化：**在池化后的第5层输出上训练回归网络

**检测：**略

**累积预测：**



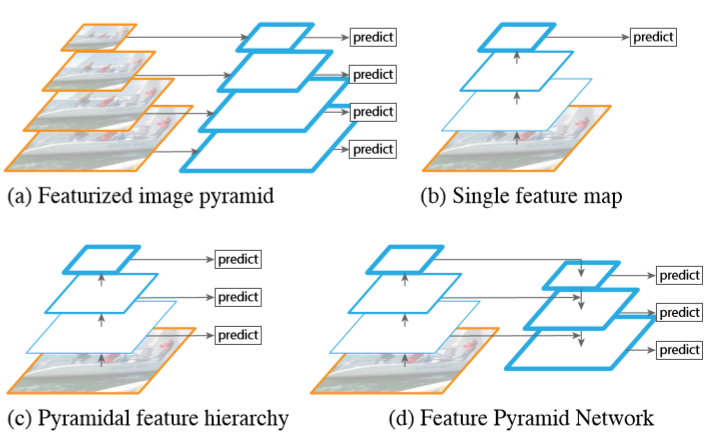
## 2017 FPN

CVPR - Feature Pyramid Networks for Object Detection

<https://www.jianshu.com/p/5a28ae9b365d>

多维度的目标检测一直以来都是通过将缩小或扩大后的不同维度图片作为输入来生成出反映不同维度信息的特征组合。这种办法确实也能有效地表达出图片之上的各种维度特征，但却对硬件计算能力及内存大小有较高要求。

FPN提出了一种可有效在单一图片视图下生成对其的多维度特征表达的方法。它可以有效地赋能常规CNN模型，从而可以生成出表达能力更强的feature maps以供下一阶段计算机视觉任务使用。本质上说它是一种加强主干网络CNN特征表达的方法。

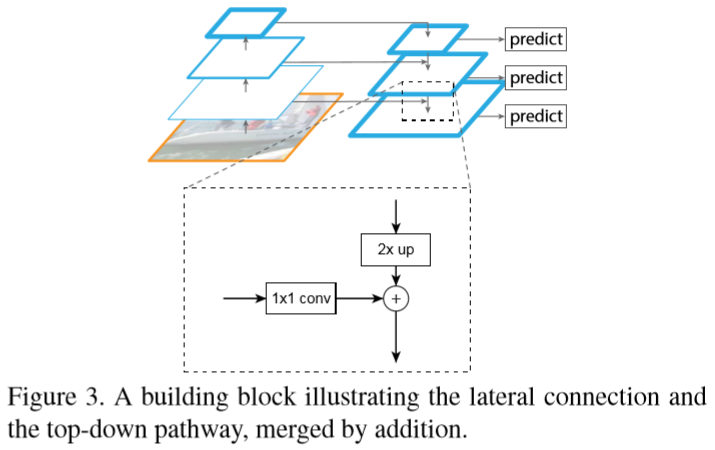


(a)即为常规的生成一张图片的多维度特征组合的经典方法。即对输入图片通过压缩或放大形成不同维度的图片，使用同一模型对这些不同维度的图片分别处理后，再将这些分别得到的特征（feature maps）组合起来。此种方法缺点在于对计算机的算力及内存大小都有较高要求。

(b)只拿单一维度的图片做为输入，然后经CNN模型处理后，拿最终一层的feature maps作为最终的特征集。此种方法只能得到单一维度的信息。优点是计算简单。对小维度的目标检测性能不是很好。

(c)同样是拿单一维度的图片做为输入，不过最终选取特征组合时，不只选用了最后一层的high level feature maps，同样选用稍靠下的反映图片low level信息的feature maps。然后将这些不同层次（反映不同level的图片信息）的特征简单合并起来（一般为concat处理），用于最终的特征组合输出。此方法可见于SSD当中。不过SSD在选取层特征时都选用了较高层次的网络，于是它对更小维度的目标检测效果就不大好。

(d)同(c)有些类似，也是拿单一维度的图片作为输入。它会选取所有层的特征来处理然后再联合起来做为最终的特征输出组合。还对这些反映不同级别信息的各层自上向下进行了再处理从而形成较好的特征表达。



**基本结构：**

1. 自下至上的通路（Bottom-top pathway）：普通CNN特征自底至上逐层浓缩表达特征的一个过程。较底的层反映较浅层次的图片信息特征像边缘等；较高的层则反映较深层次的图片特征像物体轮廓、类别等；
2. 自上至下的通路（Top-bottome pathway）：上采样。横向连接（lateral connections）将上采样的结果和自底向上生成的相同大小的feature map进行融合（merge）。在融合之后还会再采用3\*3的卷积核对每个融合结果进行卷积，目的是消除上采样的混叠效应（aliasing effect）

## 2017 RetinaNet & Focal loss

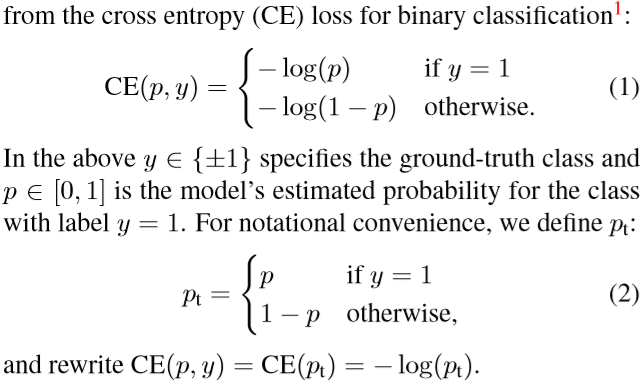
ICCV - Focal Loss for Dense Object Detection

<https://www.jianshu.com/p/8e501a159b28>

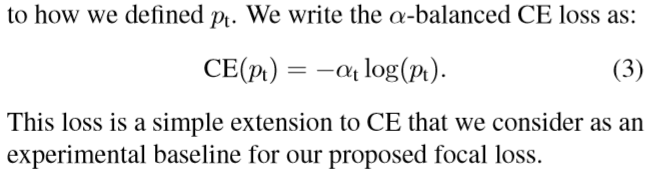
常规的单阶段目标检测网络像SSD在模型训练时会先大密度地在模型终端的feature maps上生成目标候选区域。然后再分别对这些候选区域进行分类与位置回归识别。在这些候选区域中，绝大多数都是不包含待检测目标的图片背景，这样就造成了机器学习中经典的训练样本正负不平衡的问题。

常用的解决此问题的方法就是负样本挖掘，或其它更复杂的样方法。此篇文章提出了可通过候选区域包含潜在目标概率进而对最终的training loss进行较正的方法。实验表明这种新提出的focal loss在单阶段目标检测任务上表现突出，有效地解决了此领域里面潜在的类别不平衡问题。

**CE(cross-entropy) loss：**

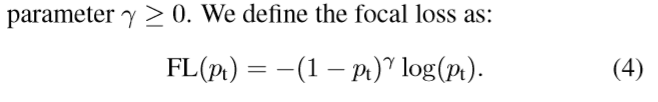


**Balanced CE loss：**



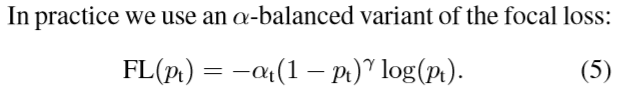
考虑到上节中提到的类别不平衡问题对最终training loss的不利影响，我们自然会想到可通过在loss公式中使用与目标存在概率成反比的系数对其进行较正。如下公式即是此朴素想法的体现。它也是作者最终Focus loss的baseline。

**Focal loss：**



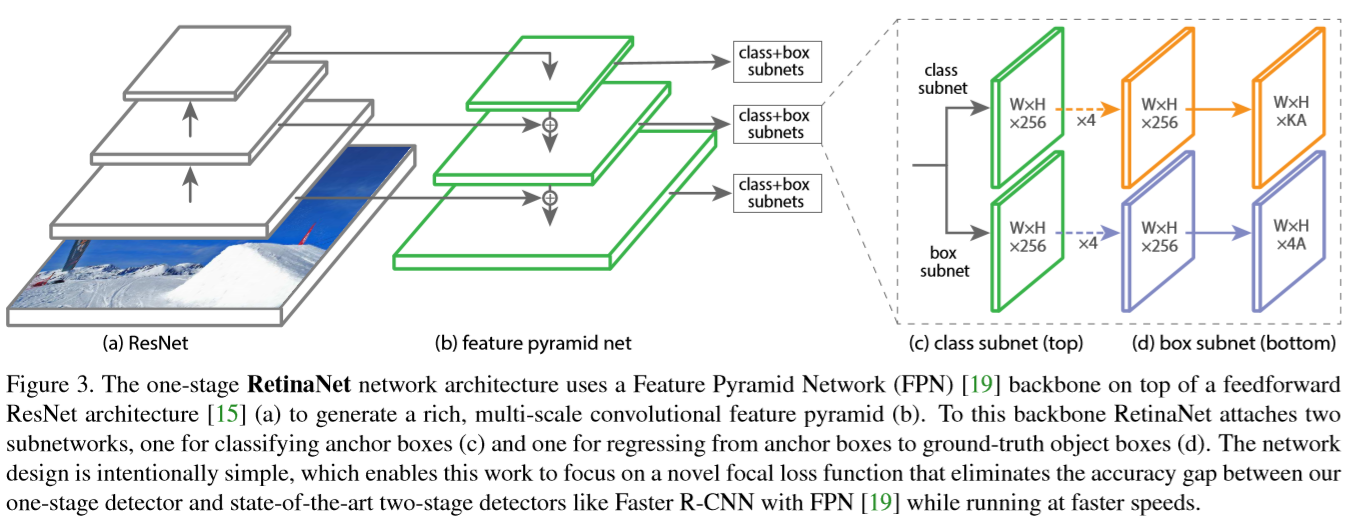
focal loss所加的指数式系数可对难易样本对loss的贡献自动调节。当某样本类别比较明确些，它对整体loss的贡献就比较少；而若某样本类别不易区分，则对整体loss的贡献就相对偏大。这样得到的loss最终将集中精力去诱导模型去努力分辨那些难分的目标类别，于是就有效提升了整体的目标检测准度。

我们把这两种单独的改进进行合并，最终Focal Loss的形式为：



**RetinaNet检测框架**

RetinaNet本质上是Resnet + FPN + 两个FCN子网络。



主干网络：resnet-50或resnet-101

FPN：resnet-50

在FPN输出的feature maps集合上，分别使用了两个FCN子网络（它们有着相同的网络结构却各自独立，并不share参数）用来完成目标框类别分类与位置回归任务。

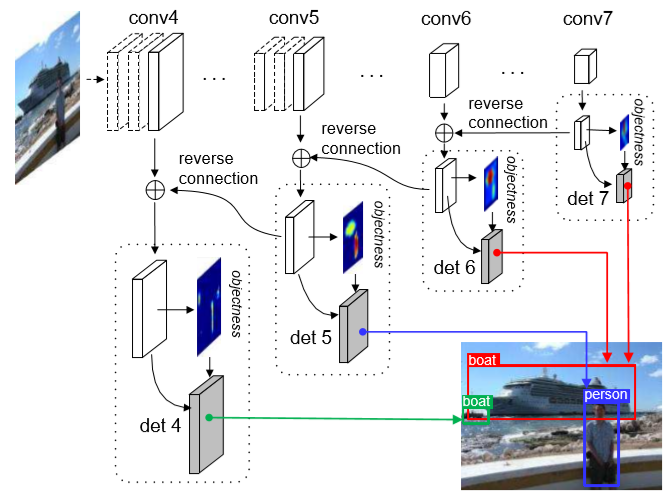
# Merge

## 2017 RON

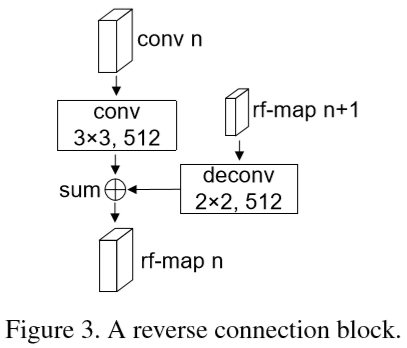
CVPR - Ron: Reverse connection with objectness prior networks for object detection

属于one-stage和two-stage结合的方法，关注于两个目标检测领域的基本问题：

1. 多尺度目标定位：研究设计了反向连接（reverse connection），使网络能够检测多层 CNN中的对象。
2. 负样本挖掘：提出了目标先验（objectness prior）来引导目标对象搜索；利用多任务损失函数优化整个网络，这样网络就能直接预测最终检测结果。



**反向连接：**



将细粒度的信息和高度抽象的信息结合可以有助于多尺度目标检测。受残差（residual connection）启发，在传统CNN结构上提出了反向连接（reverse connection），这使得前向特征能具有更多的语义信息。

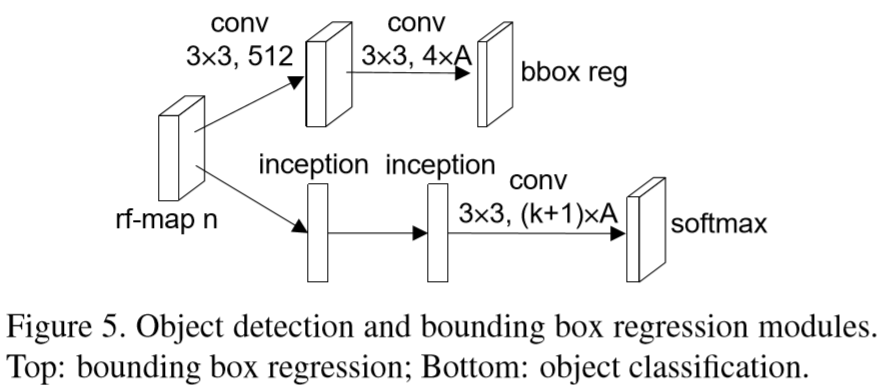
**Anchor：**

4个scale的特征图。第k个特征图上anchor的尺度为{2k-1, 2k}，基础尺度为原图的1/10。每个尺度上5个anchor，宽高比为{1/3, 1/2, 1, 2, 3}。每个特征图的每个位置上有10个anchor。

**目标先验：**

只有一小部分的Anchor真正包含物体。基于区域（two-stage）的方法通过RPN来克服这个问题，而RON添加了目标先验（objectness prior）来指导目标的搜索，没有生成新的区域。具体来说，在Softmax后添加一个3×3×2的卷积核来指示在每个参考框中是否存在物体。

**分类与回归：**

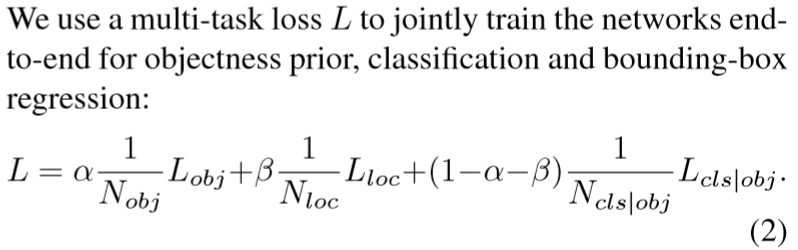


**训练策略：**

每一个参考框有两个label，一个是不是目标的label，另一个类别label。对每个ground truth box，找到和它重叠面积最大的候选区域。对每个候选区域，找到和它重叠面积大于0.5的真实框。这种匹配策略保证了每一个真实框都至少有一个候选框和它对应。对于重叠面积小于0.3的，分配负样本label。

前向传播时，网络首先产生目标先验，并进行类别检测。反向传播时，只有objectness得分大于某个阈值的anchor被用于更新detection网络的参数。额外的计算仅仅在于为反向传播选择训练样本。

**损失函数：**

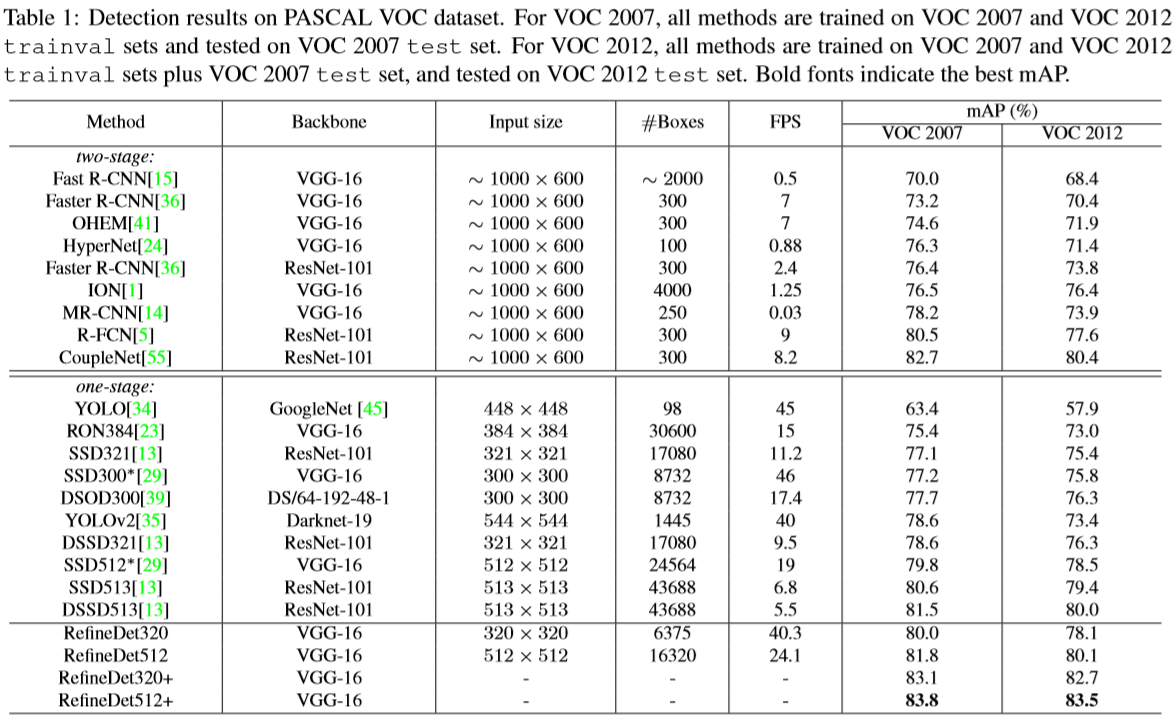


包括anchor先验的loss，Softmax；位置回归loss，L1；分类loss，Sotfmax；

**实验结果：**

**速度：**384\*384，15FPS。比two-stage和SSD快一点。

**精度：（引用**RefineDet的图**）**



个人评价：two-stage是用一个比较复杂的方法生成区域提案；RON是用一个非常简单的方法生成区域提案，再用一个比较简单的方法过滤这些提案。在精度和速度上有所提高。

## 2018 RefineDet

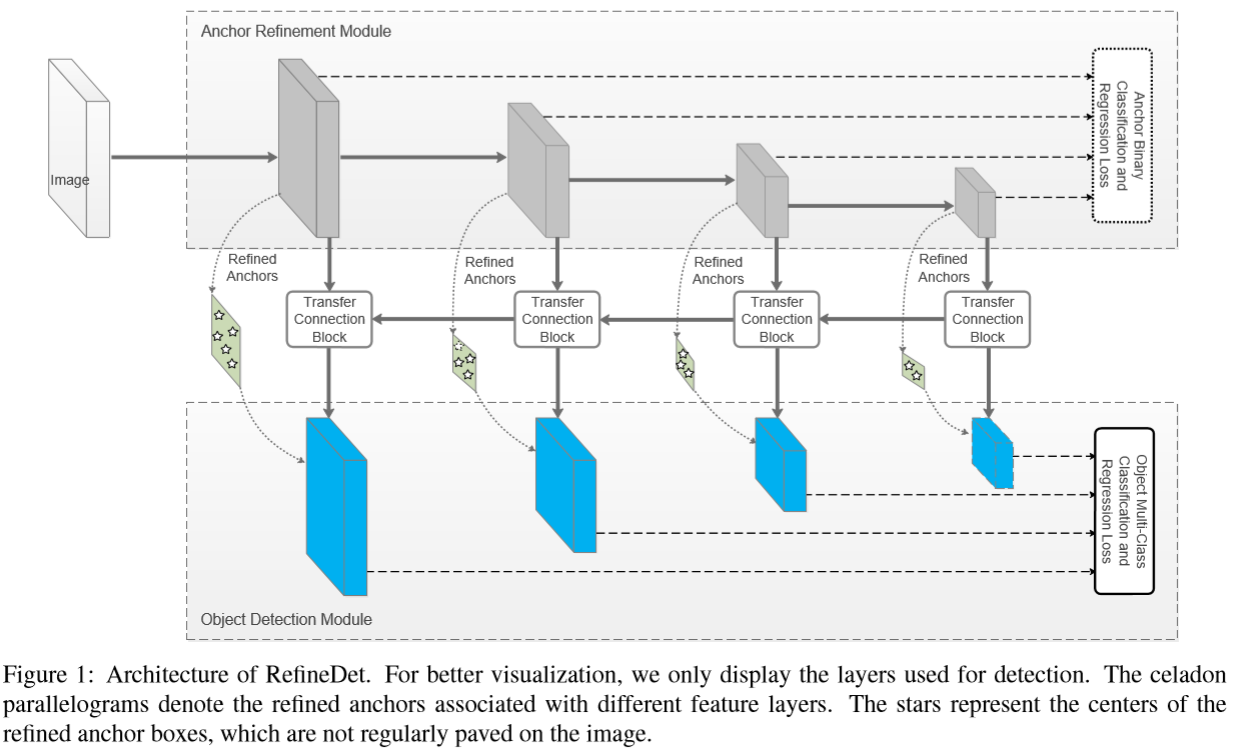
CVPR - Single-Shot Refinement Neural Network for Object Detection

<https://blog.csdn.net/nwu_nbl/article/details/81110286>

引入two stage类型的object detection算法中对box的由粗到细的回归思想（先通过RPN网络得到粗粒度的box信息，然后再通过常规的回归支路进一步回归得到更精确的框信息，这也是two stage类型的object detection算法效果优于one stage类型的重要原因）。

引入类似FPN网络的特征融合操作用于检测网络，可以有效提高对小目标的检测效果，检测网络的框架还是SSD。

RefineDet由两个相互连接的模块组成：anchor refinement module和object detection module。同时设计transfer connection block来传输ARM的特征。Multi-task loss function使得能够以端到端的方式训练整个网络。



**ARM：**

识别并删除negative anchors，以减少分类器的搜索空间；粗略调整anchors的位置和大小，为后续提供更好的初始化回归。这部分类似Faster R-CNN算法中的RPN网络。

**ODM：**

采用ARM产生的refined anchors作为输入，进一步改善回归和预测多类标签。这部分基本上是SSD，也是融合不同层的特征，然后做multi class classification和regression。

和FPN算法类似，这里的浅层feature map（size较大的蓝色矩形块）融合了高层feature map的信息，然后预测bbox是基于每层feature map（每个蓝色矩形块）进行，最后将各层结果再整合到一起。而在SSD中浅层的feature map是直接拿来用的，并没有和高层的feature map融合。这样做的好处就是对小目标物体的检测效果更好，这在FPN和RON等算法中已经证明过了。

**TCB：**

将ARM中的特征传输到ODM中用以预测位置、大小和类别标签。这部分是做特征的转换操作，和FPN算法的特征融合很像。