2019

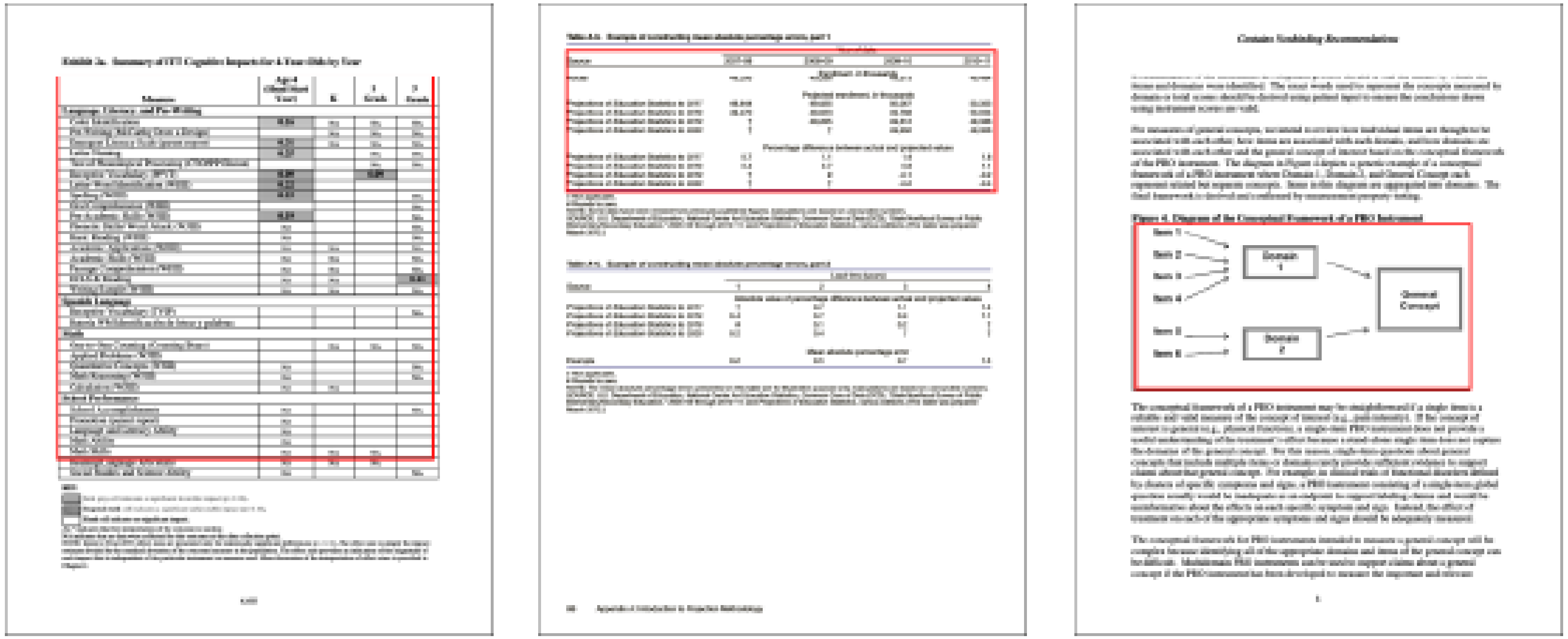
表格检测介绍

汇报人：宁子鑫　时间：2020.5.4

目录
CONTENTS

任务概览

- 检测问题，检测图片中表格位置



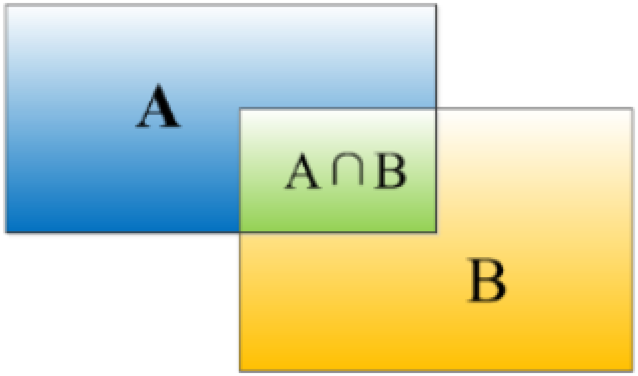
(a) (b) (c)

任务概览-评价标准

- **mAP** mean Average Precision （平均准确度均值）
- Precision: TP / (TP + FP)
- Recall: TP / (TP + FN)

TP: IoU>0.5
FP: IoU<=0.5

重叠度（**IOU**）：



IOU=(A∩B)/(A∪B)

任务概览-数据集简介

- **ICDAR 2013**
- **ICDAR 2017**
- **ICDAR 2019 (最新)**
- **TableBank**
  - 417,234张图片，采用word 和 latex生成

```
\begin{table}[]
    \centering
    \setlength{\fboxsep}{1pt}
    \fcolorbox{bordercolor}{white}{
    \begin{tabular}{}
        ...
    \end{tabular}}
\end{table}
```

两步方法-RCNN

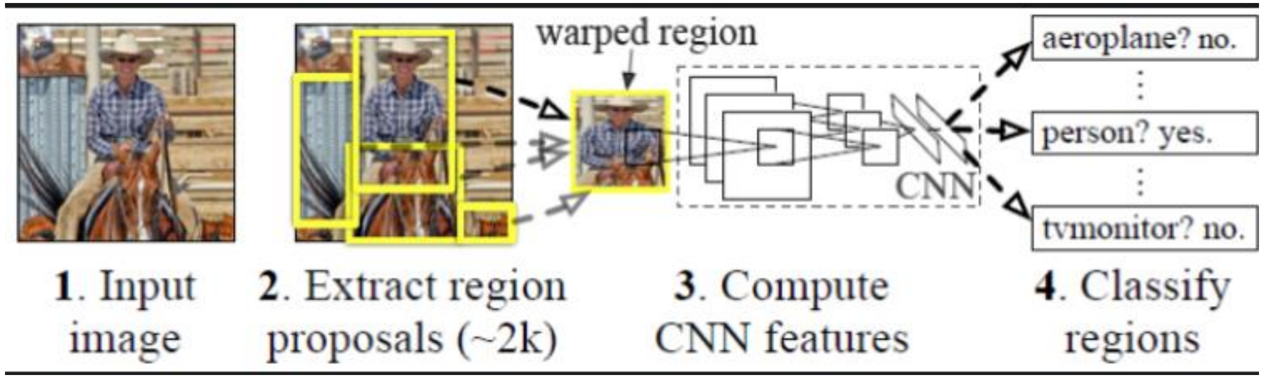
Rich feature hierarchies for accurate object detection and semantic segmentation

1.候选区域生成： 一张图像生成1K~2K个候选区域 （采用Selective Search 方法）

2.特征提取： 对每个候选区域，使用深度卷积网络提取特征 （CNN）

3.类别判断： 特征送入每一类的SVM 分类器，判别是否属于该类

4.位置精修： 使用回归器精细修正候选框位置



两步方法-RCNN

Rich feature hierarchies for accurate object detection and semantic segmentation

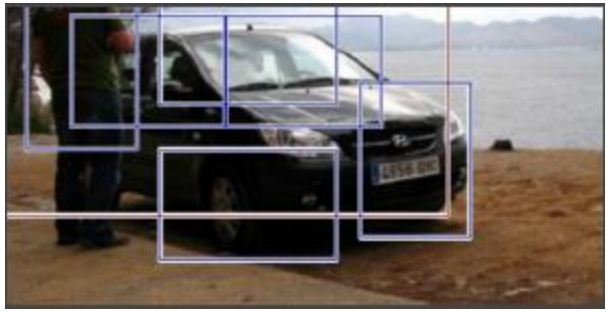
问题:
 Selective Search搜出的候选框是矩形的，而且是大小各不相同。
如何放入CNN中？
 (1)各向异性缩放
 (2)各向同性缩放

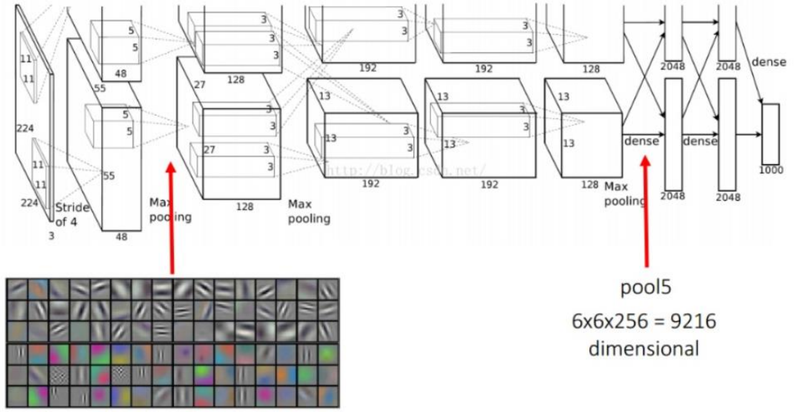
Alexnet预训练后 fine-tuning

提取propasals > region proposals归一化 > CNN特征提取 >SVM分类器打分 >非极大值抑制（NMS）> 输出bbox

非极大值抑制

1)从最大概率矩形框F开始，分别判断A~E与F的重叠度IOU是否大于某个设定的阈值;
2)假设B、D与F的重叠度超过阈值，那么就扔掉B、D；并标记第一个矩形框F，是我们保留下来的。
3)从剩下的矩形框A、C、E中，选择概率最大的E.
4)重复1

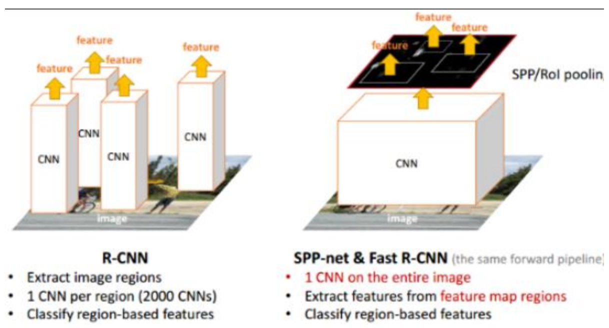




两步方法-SPPNet

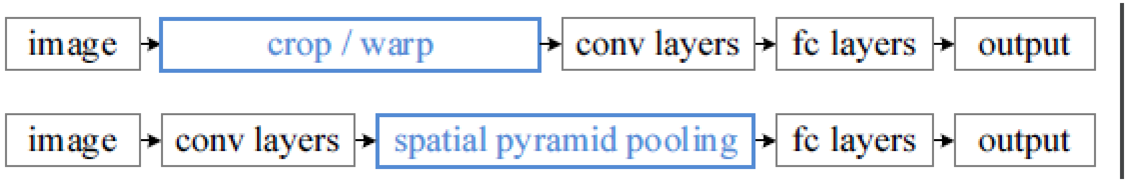
Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition

SPP-net提出：能否在feature map上提取ROI特征，这样就只需要在整幅图像上做一次卷积



问题：

ROI的在特征图上的对应的特征区域的维度不满足全连接层的输入要求怎么办？



Selective Search得到候选区域->CNN提取ROI特征->类别判断->位置精修

两步方法-SPPNet

Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition

特殊池化手段：空间金字塔池化

不同尺寸的图像也可以使 池化n 产生固定的 输出维度。

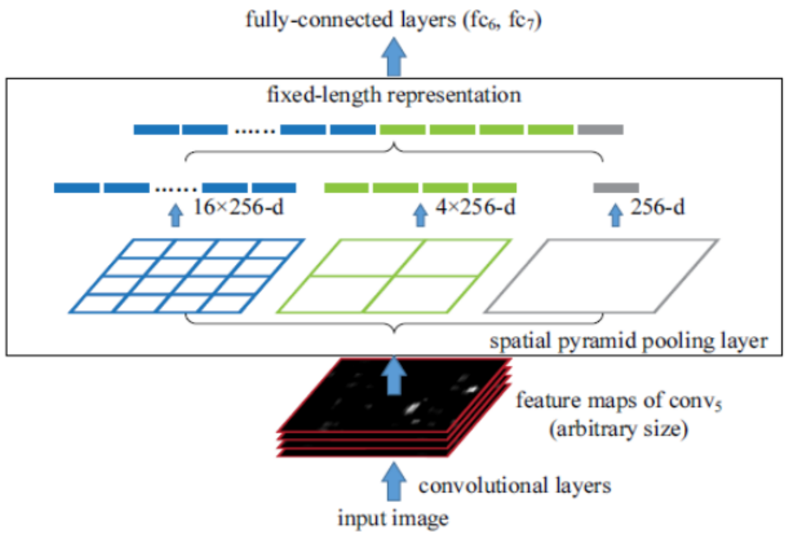


Figure 3: A network structure with a **spatial pyramid pooling layer**. Here 256 is the filter number of the $conv_5$ layer, and $conv_5$ is the last convolutional layer.

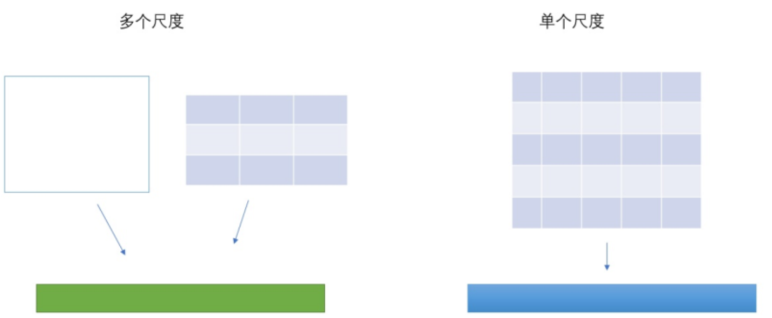
SPPNet将比较耗时的卷积计算对整幅图像只进行一次，之后使用SPP将窗口特征图池化为一个固定长度的特征表示。

两步方法-Fast R-CNN

Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition

1 实现大部分end-to-end训练(提proposal阶段除外)

2 提出了一个RoI层，算是SPP的变种，SPP是pooling成多个固定尺度，RoI只pooling到单个固定的尺度



Bounding-box Regression

输入的box坐标也放到深度神经网络里然后进行一些优化

$$t^* = (t^*_x, t^*_y, t^*_w, t^*_h),$$

$$t = (t_x, t_y, t_w, t_h)$$

$$L_{loc}(t, t^*) = \sum_{i \in \{x,y,w,h\}} \mathrm{smooth}_{L_1}(t_i, t^*_i)$$

其中

$$\mathrm{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & |x| \le 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

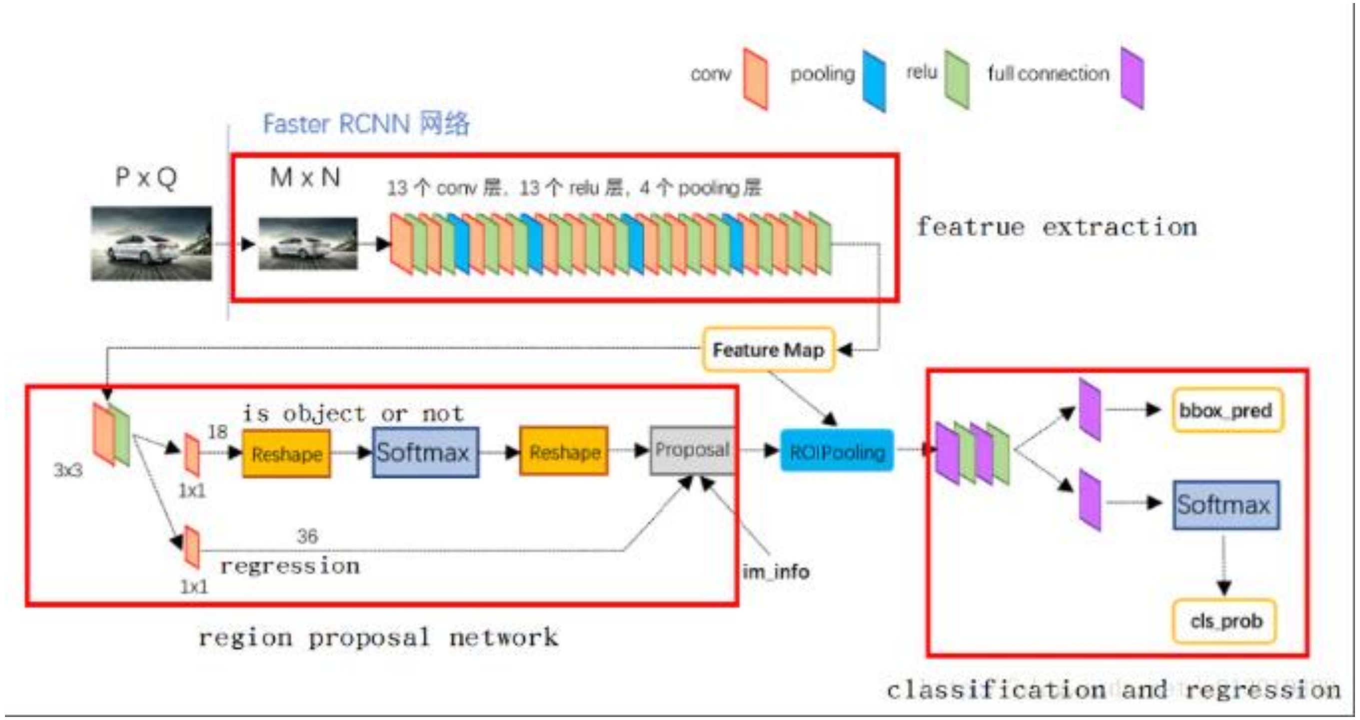
两步方法-Faster R-CNN

Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition

把proposal阶段也用CNN实现

网络中的各个卷积层特征（feature map）也可以用来
预测类别相关的region proposal（不需要事先执行诸如selective search之类的算法）

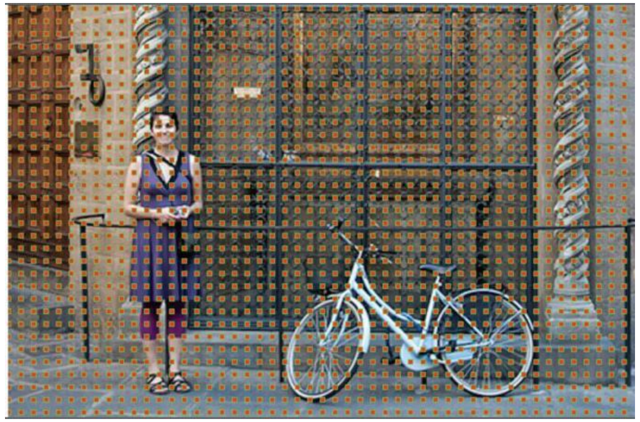
区域生成网络 （ Region Proposal Networks ）



两步方法-Faster R-CNN

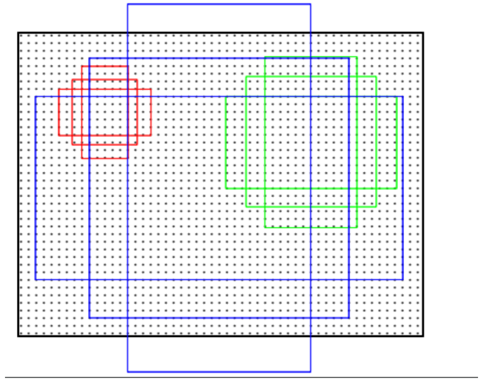
RPN

用CNN来生成候选窗口，通过得分排序等方式挑出量少质优的框



Anchor

3个scale（三种面积[公式]），3个aspect ratio({1:1,1:2,2:1}) ——9个框



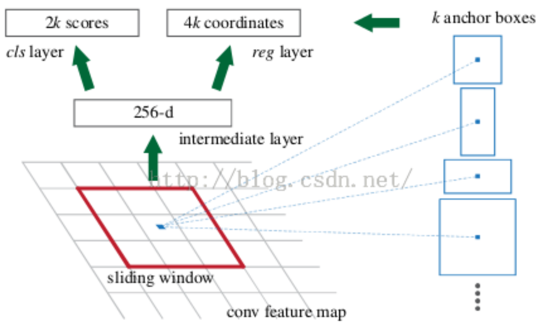
a. 对每个标定的ground true box区域，与其重叠比例最大的anchor记为 正样本 (保证每个ground true 至少对应一个正样本anchor)

b. 如果其与任意一个标定的重叠比例都小于0.3，记为负样本

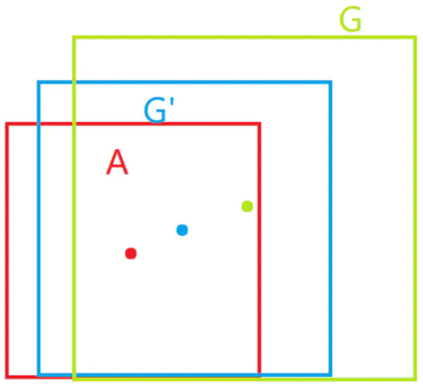
c. 对a),b)剩余的anchor，弃去不用

d. 跨越图像边界的anchor弃去不用

两步方法-Faster R-CNN



左边支路用来分类前背景，
右边支路用来回归框的位置



G代表真实框，A代表anchor给出的框，我们的目的是让A更接近G

$$
\begin{aligned}
t_{\mathrm{x}} &= (x - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t_{\mathrm{y}} = (y - y_{\mathrm{a}})/h_{\mathrm{a}}, \\
t_{\mathrm{w}} &= \log(w/w_{\mathrm{a}}), \quad t_{\mathrm{h}} = \log(h/h_{\mathrm{a}}), \\
t^{*}_{\mathrm{x}} &= (x^{*} - x_{\mathrm{a}})/w_{\mathrm{a}}, \quad t^{*}_{\mathrm{y}} = (y^{*} - y_{\mathrm{a}})/h_{\mathrm{a}}, \\
t^{*}_{\mathrm{w}} &= \log(w^{*}/w_{\mathrm{a}}), \quad t^{*}_{\mathrm{h}} = \log(h^{*}/h_{\mathrm{a}}),
\end{aligned}
$$

RPN的总体Loss函数：

$$
L(\{p_i\}\{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)
$$

两步方法-Table Detection using Deep Learning 2017 ICDAR

- Image Transformation
- Faster R-CNN

对图像的蓝、绿、红通道进行欧氏距离变换
、线性距离变换和最大距离变换

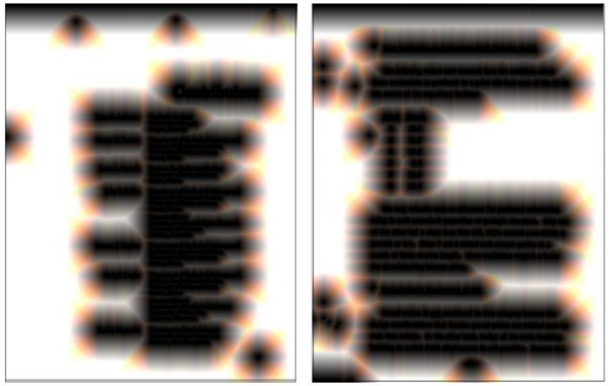
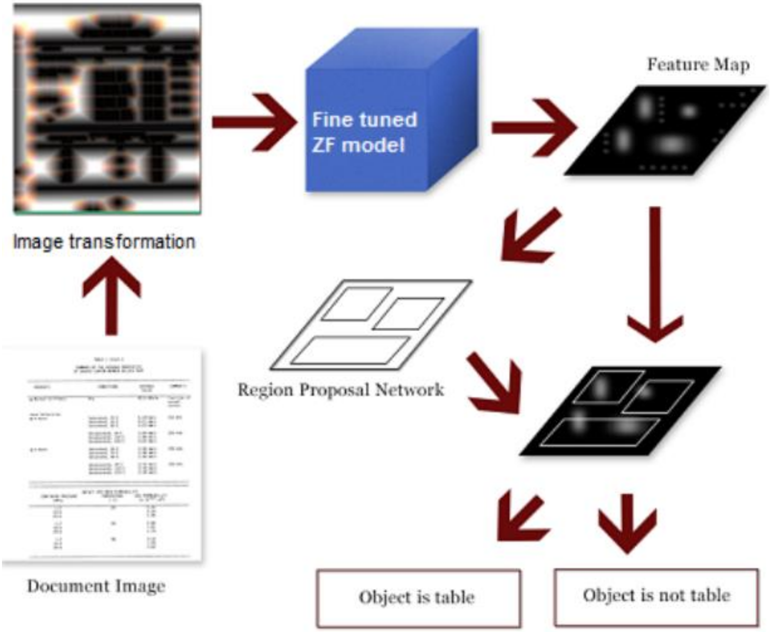


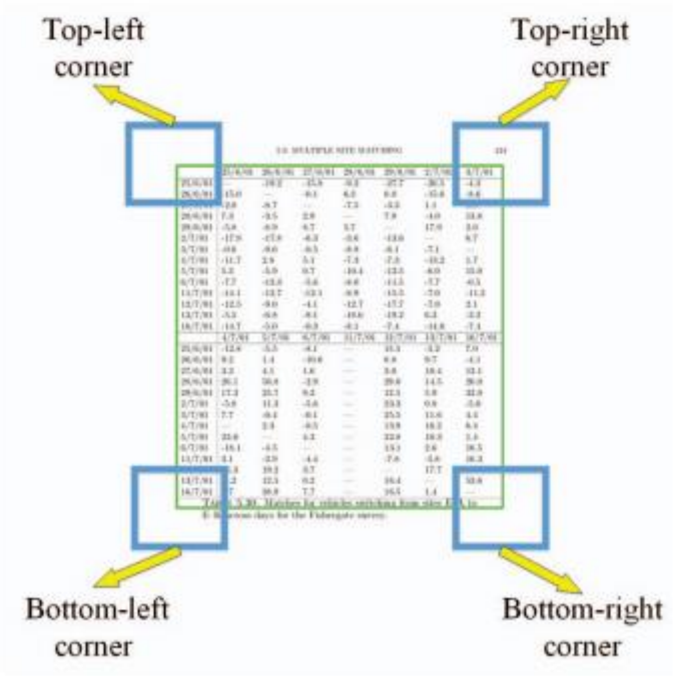
Fig. 1: Transformed Images

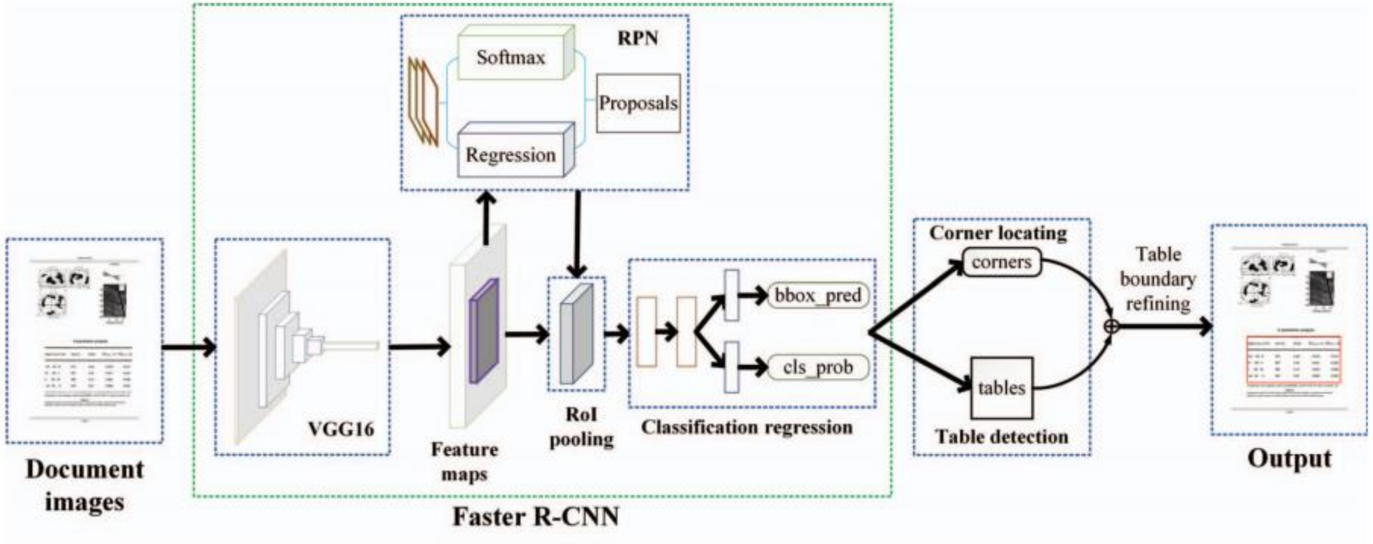


Area Precision	93.2	95.0	84.5	82.3
Area Recall	64.29	64.3	89.17	90.67
F1 Score	76.09	76.69	86.77	86.29

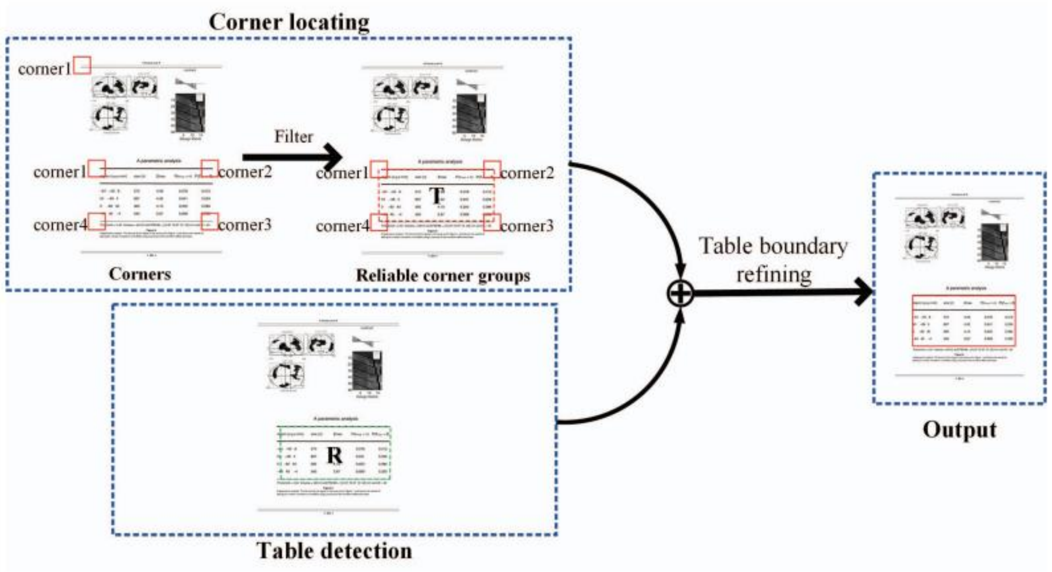
两步方法-Faster R-CNN Based Table Detection Combining Corner Locating 2019 ICDAR

在Faster R-CNN基础上使用了“角点”来提升表格检测的精准度





两步方法-Faster R-CNN Based Table Detection Combining Corner Locating 2019 ICDAR



C1和C2大致在同一水平线上。

•C3和C4大致在同一水平线上。

•C1和C4大致在同一条垂直线上。

C2和C3大致在同一条垂直线上。

Method Head	Precision	Recall	F-measure
SSD [12]	0.071	0.959	0.132
CNN [17]	0.230	0.221	0.225
Faster R-CNN(VGG_CNN_M_1024 [11]	0.670	0.940	0.782
Faster R-CNN+edge_based information [11]	0.842	0.890	0.865
Faster R-CNN(VGG16)	0.924	0.918	0.921
Our method	0.943	0.956	0.949
Faster R-CNN+CCs+CRFs [11]	0.968	0.953	0.960

单步方法-Yolo V1

直接在输出层回归bounding box的位置和bounding box所属的类别

整张图作为网络的输入，把 Object Detection 的问题转化成一个 Regression 问题

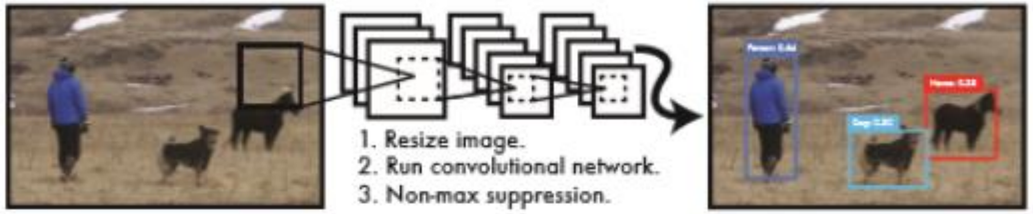
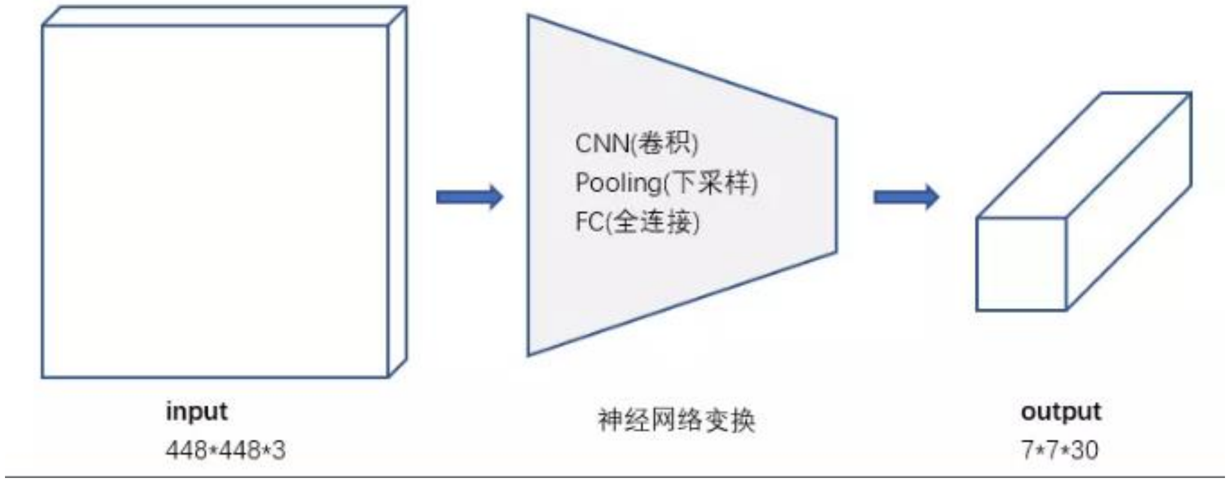
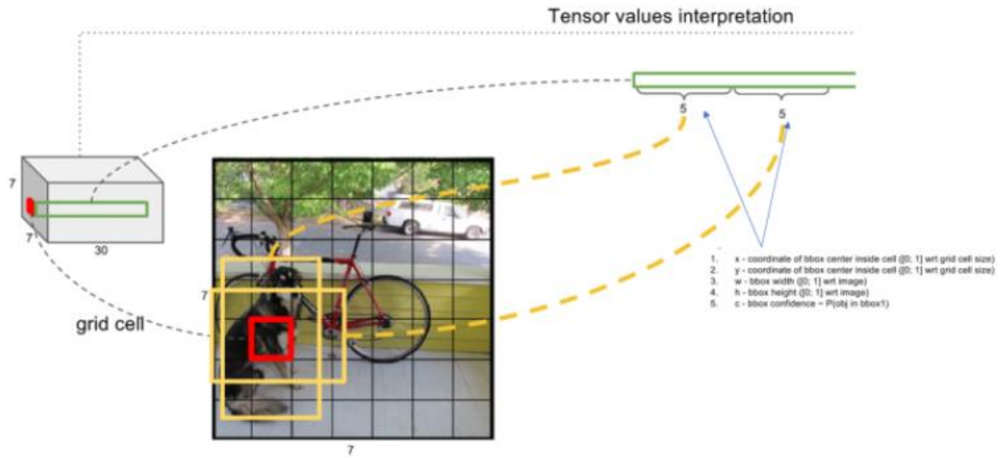


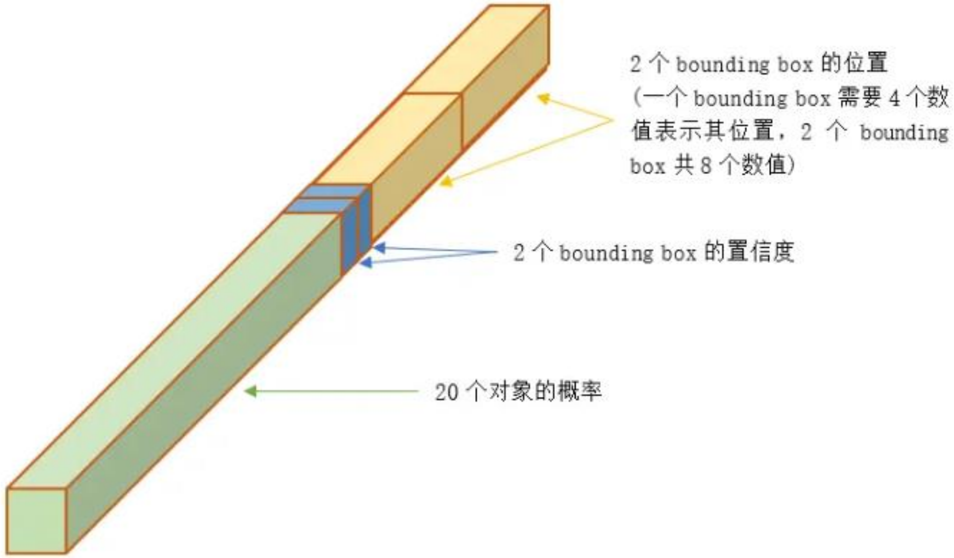
Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448 × 448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.



单步方法-Yolo V1



每个bounding box需要4个数值来表示其位置，
(Center_x,Center_y,width,height)



每个网格还要预测类别信息

预测2个 bounding box 和 20个类别概率，
输出就是 7x7x(5x2 + 20)

单步方法-Yolo V1

损失函数：

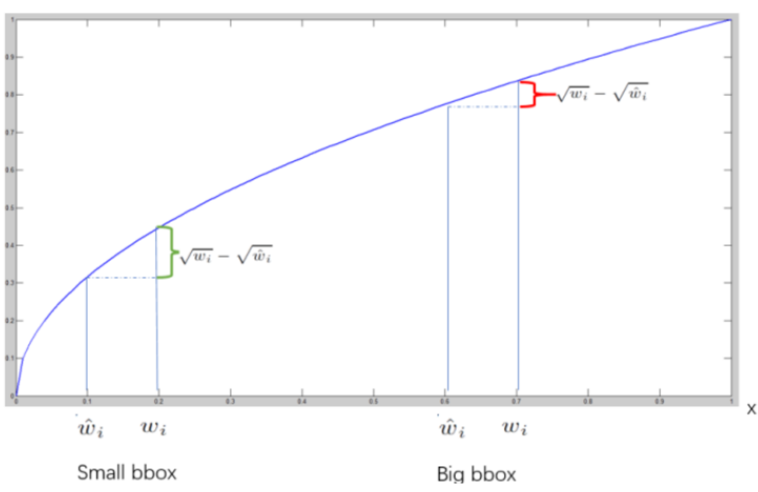
$$\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$ 边框中心点误差

$$+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$ 边框宽度、高度误差

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$ 置信度误差(边框内有对象)

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$ 置信度误差(边框内无对象)

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$ 对象分类误差



$\lambda_{\textbf{coord}}$ 取值为5

λ_{noobj} 取值为0.5

单步方法-Yolo V1

每个网格有：20个对象的概率*2个bounding box的置信度，
共40个得分（候选对象）

$$Score_{ij} = P(C_i|Object) * Confidence_j$$

每种对象有 98个得分。遍历所有对象，分别 进行NMS

单步方法-Yolo V2　cvpr 2017

YOLO2主要有两个大方面的改进：

1.使用一系列的方法对YOLO进行了改进，在保持原有速度的同时提升精度得到YOLOv2。

1）Batch Normalization

2）High Resolution Classifier

3）Convolutionlal With Anchor Boxes

4）Dimension Clusters

2.提出了一种目标分类与检测的联合训练方法，同时在COCO和ImageNet数据集中进行训练得到YOLO9000，实现9000多种物体的实时检测。

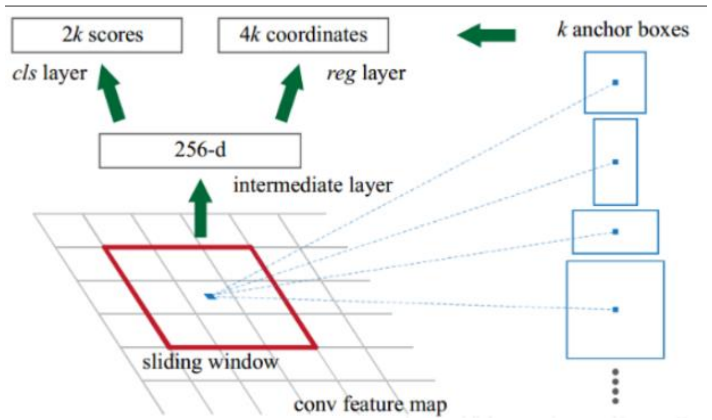
单步方法-Yolo V2

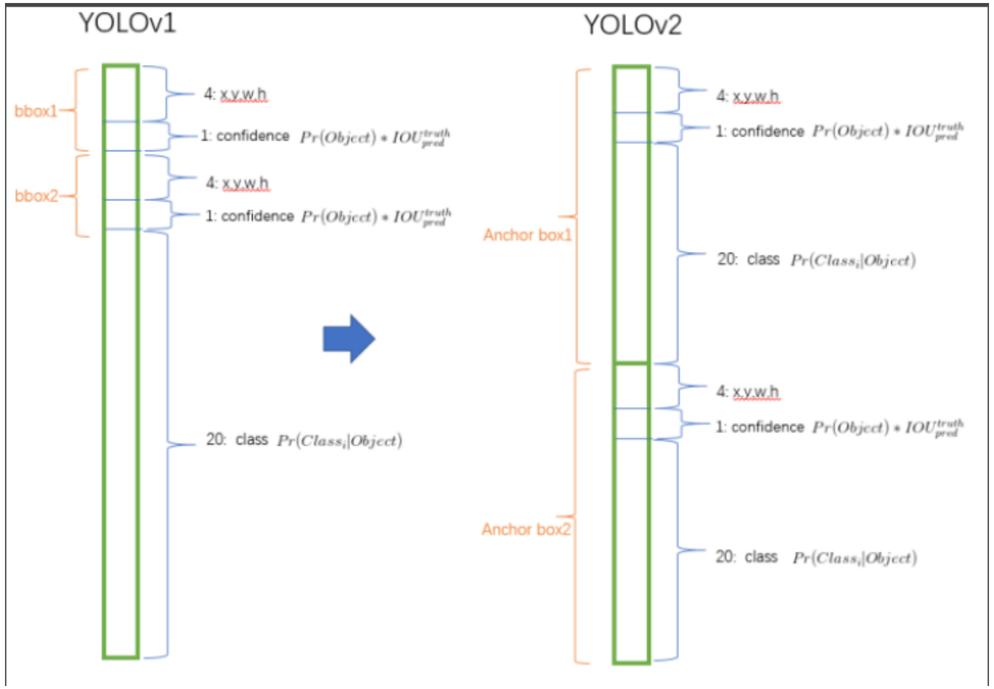
Convolutional With Anchor Boxes.

1.移除全连接层（以获得更多空间信息）使用 anchor boxes 取预测 bounding boxes

2.缩减网络，让图片输入分辨率为416 * 416

3.由anchor box同时预测类别和坐标





单步方法-Yolo V2

Dimension Clusters

Faster-RCNN中anchor boxes的个数和宽高维度往往是手动精选的,能否通过某种算法选择

K-means聚类方法，通过对数据集中的ground true box做聚类

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

Direct location prediction

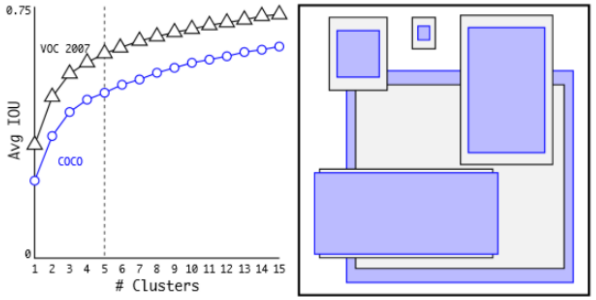
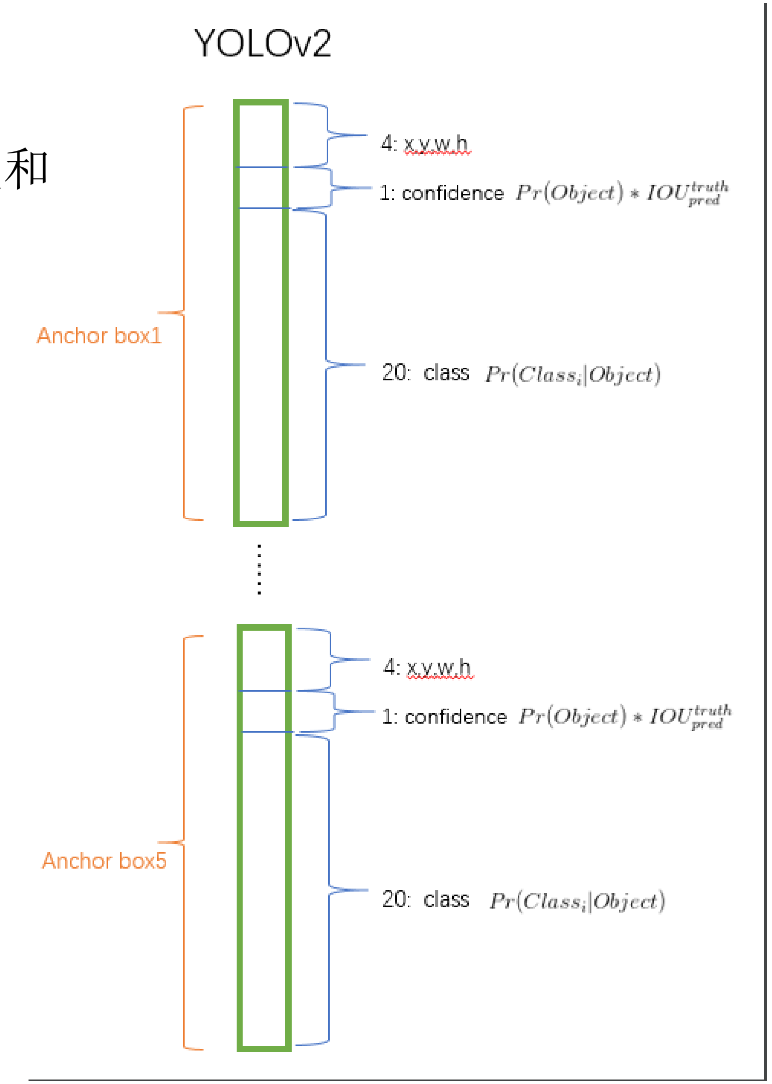


Figure 2: Clustering box dimensions on VOC and COCO. We run k-means clustering on the dimensions of bounding boxes to get good priors for our model. The left image shows the average IOU we get with various choices for k. We find that $k = 5$ gives a good tradeoff for recall vs. complexity of the model. The right image shows the relative centroids for VOC and COCO. Both sets of priors favor thinner, taller boxes while COCO has greater variation in size than VOC.

单步方法-Yolo V2

对于VOC数据集，预测5种boxes，每个box包含5个坐标值和20个类别，所以总共是5 * （5+20）= 125个输出维度。



单步方法-A YOLO-based Table Detection Method 2019 icdar

首次将yolov3 应用在table detection

一种锚优化方法

两种后处理方法

K= 9

$$D(box, centroid) = 1 - IoU(box, centroid) \qquad (1)$$

后处理：

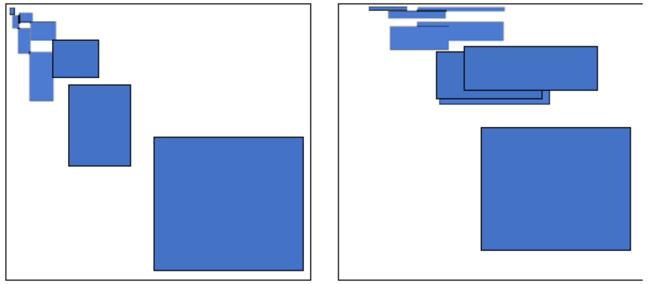
1.删除预测区域的空白

2.如果违反某些规则 则视为非表

1）它与页面的顶部或底部之间的最小距离小于0.05×页面高度

2）它的面积小于500像素

3）它的宽比高或者高比宽大于12。



(a) Original anchor sizes　　(b) Optimized anchor sizes

Dataset	IoU Threshold	Model	Precision	Recall	F1-measure
ICDAR 2017 POD Competition, Test set, 817 images	0.6	**Ours (YOLOv3+a+p)**	**0.978**	**0.972**	**0.975**
		Li et al. [25]	0.974	0.962	0.968
		DeCNT (2018) [26]	0.965	0.971	0.968
		NLPR-PAL	0.968	0.953	0.960
		School of Software	0.934	0.940	0.937
		FastDetectors	0.903	0.940	0.921
		VisInt	0.924	0.918	0.921
		maitai-ee	0.842	0.890	0.865
		icstpku	0.857	0.773	0.813
		UITVN	0.670	0.940	0.782
		IU-vision	0.230	0.221	0.225
		HustVision	0.071	0.959	0.132
	0.8	**Ours (YOLOv3+a+p)**	**0.975**	**0.968**	**0.971**
		Li et al. [25]	0.965	0.953	0.959
		DeCNT (2018) [26]	0.967	0.937	0.952
		NLPR-PAL	0.943	0.958	0.951
		FastDetectors	0.879	0.915	0.896
		VisInt	0.829	0.823	0.826
		School of Software	0.793	0.798	0.796
		maitai-ee	0.755	0.798	0.776
		icstpku	0.804	0.726	0.763
		UITVN	0.544	0.763	0.635
		IU-vision	0.118	0.114	0.116
		HustVision	0.062	0.836	0.115
ICDAR 2013 Table Competition, Complete set, 238 images	0.5	**DeCNT (2018) [26]**	0.996	**0.996**	**0.996**
		Kavasidis et al. (2018) [21]	0.975	0.981	0.978
		Ours (YOLOv3+a+p)	**1.000**	0.949	0.973
		Sebestian et al. (2017) [23]	0.974	0.962	0.968
		Tran et al. (2015) [30]	0.952	0.967	0.958
		Hao et al. (2016) [22]	0.972	0.922	0.946

其他方法-Feature Engineering meets Deep Learning: A Case Study on Table Detection in Documents DICTA

1) Foreground Feature Engineering
2) 2) Background Feature Engineering
3) 3) Deep Learning on Encoded Features

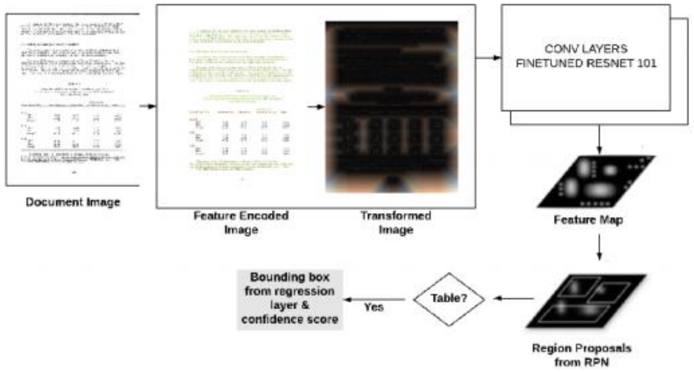


Fig. 2: Proposed approach: Features are engineered and color coded. Distance transform is applied to the original image shown in Figure 1b. Both images are added and the resultant image is be fed to the feature extractor. Feature map generated by the feature extractor is then passed to the region proposal network (RPN) which proposes the regions where tables might be present. The detection network processes the proposed regions as input and classifies them into table or non-table regions.

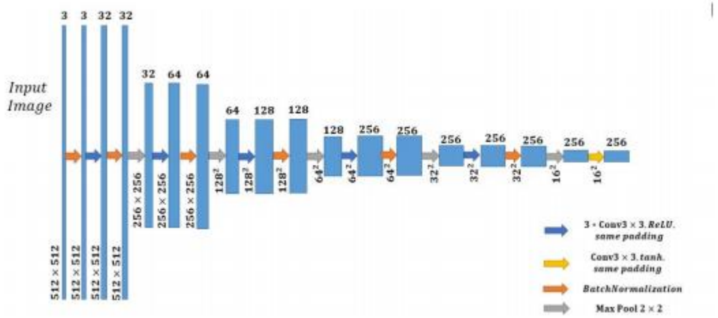
自底向上的词段聚类来提取表结构信息

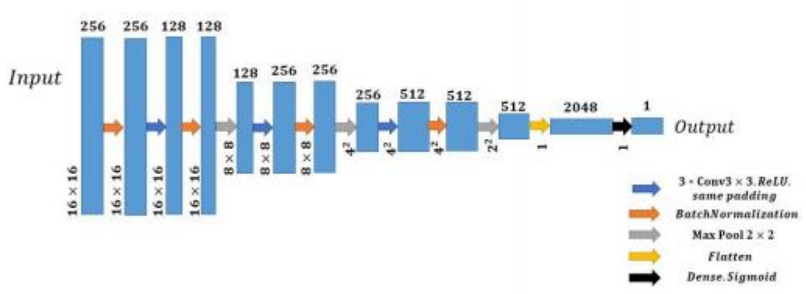
每行一个单词组成的所有块都被归类为类型1，
其他所有块都被归类为类型2

如果一个块的类型为2，并且它的宽度超过了一个定义的阈值，那么它就被标记为一个段落

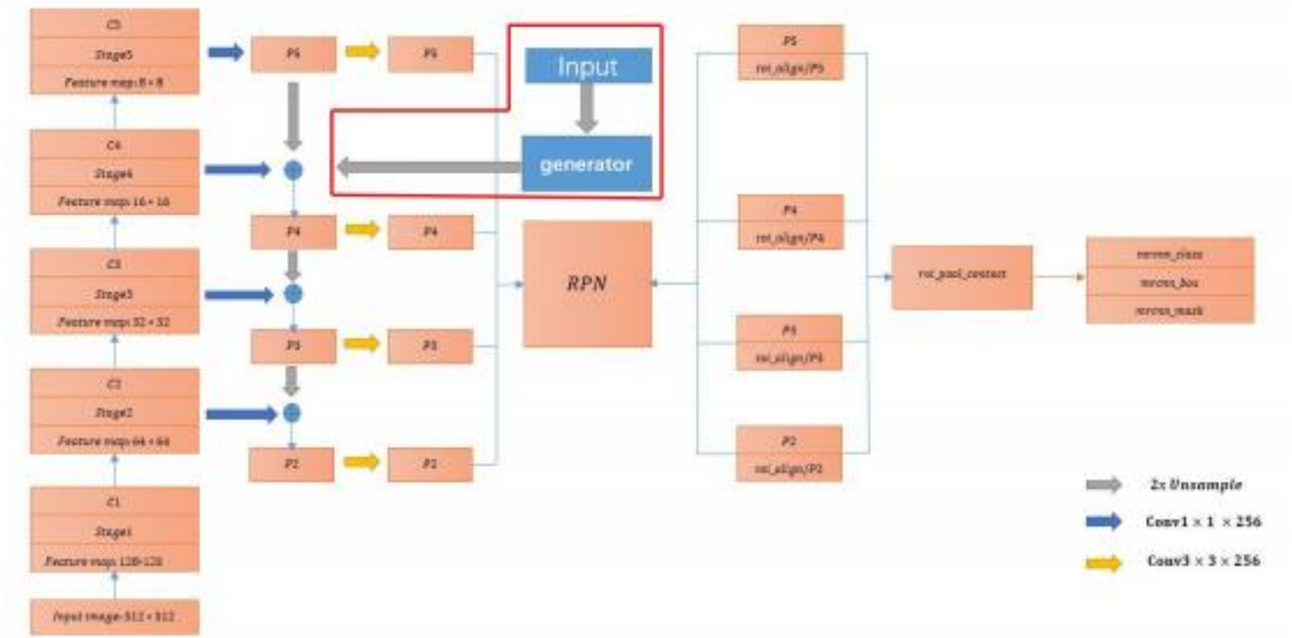
用不同的颜色编码将数字和文本信息编码在一个单一的红色通道中。数值信息的像素值为255，文本信息的像素值为128　蓝色通道包含红色和绿色通道的平均值。

其他方法-A GAN-based Feature Generator for Table Detection icdar 2019





最终将特征发生器器加入MastRcnn



1.删除表格的划线，得到虚假的图像

2.使用生成器从图像中获取特征图

3.使用真实的特征图和虚假的特征图来训练鉴别器

4.使用真实和虚假的图像训练FGAN模型

5.重复2

Model	IoU = 0.6			IoU = 0.8		
	Precision	**Recall**	**F1-measure**	**Precision**	**Recall**	**F1-measure**
U-NET	0.880	0.897	0.888	0.762	0.793	0.777
F+U-NET	**0.891**	**0.915**	**0.903**	**0.785**	**0.802**	**0.793**
MASK R-CNN	0.936	0.925	0.930	0.893	**0.908**	0.900
F+MASK R-CNN	**0.944**	**0.944**	**0.944**	**0.903**	0.903	**0.903**

2019

报告完毕 谢谢您的观看

汇报人：宁子鑫　　时间：5.4