文本检测--EAST

<Excerpt in index | 首页摘要>

Detecting Text In Natural Image With connectionist text proposal network

KeyWords Plus: CVPR, 2017

- **relevant blog**： EAST: An Efficient and Accurate Scene Text Detector
- **paper**： EAST
- **coding**： Github
- **PPT**： My_Note

  <The rest of contents | 余下全文>

指标

1、评价指标

Precision又叫查准率，**Recall**又叫查全率。这两个指标共同衡量才能评价模型输出结果。

- `TP`: 预测为1(Positive)，实际也为1(Truth-预测对了)
- `TN`: 预测为0(Negative)，实际也为0(Truth-预测对了)
- `FP`: 预测为1(Positive)，实际为0(False-预测错了)
- `FN`: 预测为0(Negative)，实际为1(False-预测错了)
  总的样本个数为: `TP+TN+FP+FN。`
- `Accuracy` = (预测正确的样本数)/(总样本数)=**(TP+TN)/(TP+TN+FP+FN)**
- `Precision` = (预测为1且正确预测的样本数)/(所有预测为1的样本数) = **TP/(TP+FP)**
- `Recall` = (预测为1且正确预测的样本数)/(所有真实情况为1的样本数) = **TP/(TP+FN)**
- `F-score` = 2*(`Precision` * `Recall`)/(`Precision` + `Recall`)

2、TensorFlow中的tf.metrics算子

深入理解TensorFlow中的tf.metrics算子

精确率（Precision）：$Precision = \frac{TP}{TP+FP}$

召回率（Recall）：$Recall = \frac{TP}{TP+FN}$

F-measure：$F - measure = \frac{2\times Precision\times Recall}{Precision+Recall}$

准确率（Accuracy）：$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$

ICDAR 2015 is used in Challenge 4 of ICDAR 2015 Robust Reading Competition [15]. It includes a total of **1500 pictures**, **1000** of which are used for **training** and the **remaining** are for **testing**.

Algorithm	Recall	Precision	F-score
Ours + PVANET2x RBOX MS*	**0.7833**	0.8327	**0.8072**
Ours + PVANET2x RBOX	0.7347	**0.8357**	**0.7820**
Ours + PVANET2x QUAD	0.7419	0.8018	0.7707
Ours + VGG16 RBOX	0.7275	0.8046	0.7641
Ours + PVANET RBOX	0.7135	0.8063	0.7571
Ours + PVANET QUAD	0.6856	0.8119	0.7434
Ours + VGG16 QUAD	0.6895	0.7987	0.7401
Yao *et al.* [41]	0.5869	0.7226	0.6477
Tian *et al.* [34]	0.5156	0.7422	0.6085
Zhang *et al.* [48]	0.4309	0.7081	0.5358
StradVision2 [15]	0.3674	0.7746	0.4984
StradVision1 [15]	0.4627	0.5339	0.4957
NJU [15]	0.3625	0.7044	0.4787
AJOU [20]	0.4694	0.4726	0.4710
Deep2Text-MO [45, 44]	0.3211	0.4959	0.3898
CNN MSER [15]	0.3442	0.3471	0.3457

Table 3. Results on ICDAR 2015 Challenge 4 Incidental Scene Text Localization task. MS means multi-scale testing.

COCO-Text is the largest text detection dataset to date. It reuses the images from **MS-COCO dataset** . A total of **63,686** images are **annotated**, in which **43,686** are chosen to be the **training** set and the rest **20,000** for **testing**.

Algorithm	Recall	Precision	F-score
Ours + VGG16	**0.324**	**0.5039**	**0.3945**
Ours + PVANET2x	0.340	0.406	0.3701
Ours + PVANET	0.302	0.3981	0.3424
Yao *et al.* [41]	0.271	0.4323	0.3331
Baselines from [36]			
A	0.233	0.8378	0.3648
B	0.107	0.8973	0.1914
C	0.047	0.1856	0.0747

Table 4. Results on COCO-Text.

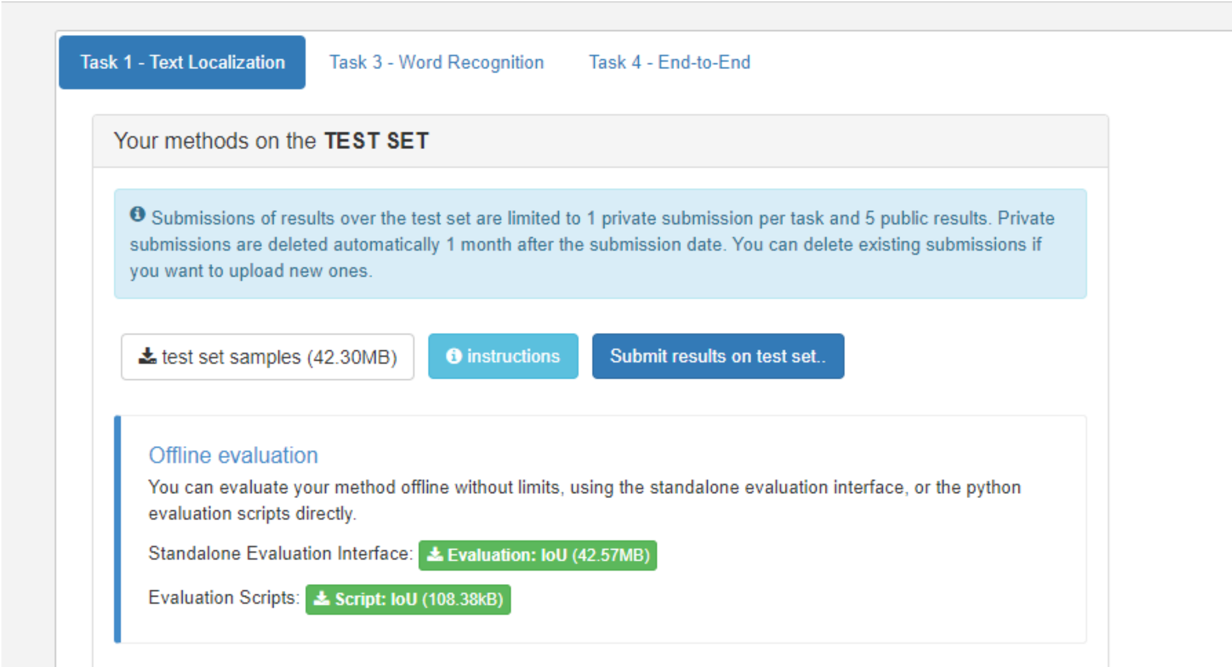
MSRA-TD500 is a dataset comprises of **300 training images** and **200 test images.** Text regions are of arbitrary orientations and annotated at sentence level. Different from the other datasets, it contains text in **both English and Chinese.**

Algorithm	Recall	Precision	F-score
Ours + PVANET2x	0.6743	**0.8728**	**0.7608**
Ours + PVANET	0.6713	0.8356	0.7445
Ours + VGG16	0.6160	0.8167	0.7023
Yao *et al.* [41]	**0.7531**	0.7651	0.7591
Zhang *et al.* [48]	0.67	0.83	0.74
Yin *et al.* [44]	0.63	0.81	0.71
Kang *et al.* [14]	0.62	0.71	0.66
Yin *et al.* [45]	0.61	0.71	0.66
TD-Mixture [40]	0.63	0.63	0.60
TD-ICDAR [40]	0.52	0.53	0.50
Epshtein *et al.* [5]	0.25	0.25	0.25

Table 5. Results on MSRA-TD500.

3、ICDAR

ICDAR 2017 Robust Reading competitions官网有专门的评价函数，可以得出**三个指标**：



预测结果：

论文中：

Algorithm	Recall	Precision	F-score
Ours + PVANET2x RBOX MS*	**0.7833**	0.8327	**0.8072**
Ours + PVANET2x RBOX	0.7347	**0.8357**	**0.7820**

实测：

Recall: 0.772267 **Precision**: 0.846437 **F-score**: 0.807653

基本网络结构

Introduction

The contributions of this work are three-fold:

- We propose a scene text detection method that consists of **two stages**: a Fully Convolutional Network and an NMS merging stage. **The FCN directly produces text regions, excluding redundant and time-consuming intermediate steps.**
- The pipeline is flexible to produce either word level or line level predictions, whose geometric shapes can be **rotated boxes or quadrangles**, depending on specific applications.
- The proposed algorithm significantly **outperforms** state-of-the-art methods in both **accuracy and speed.**

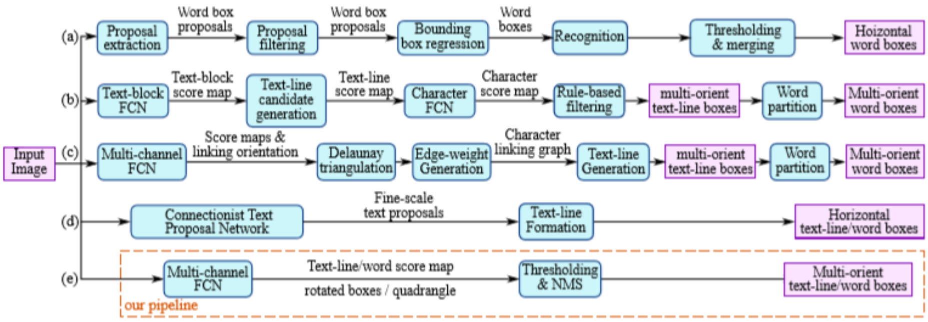
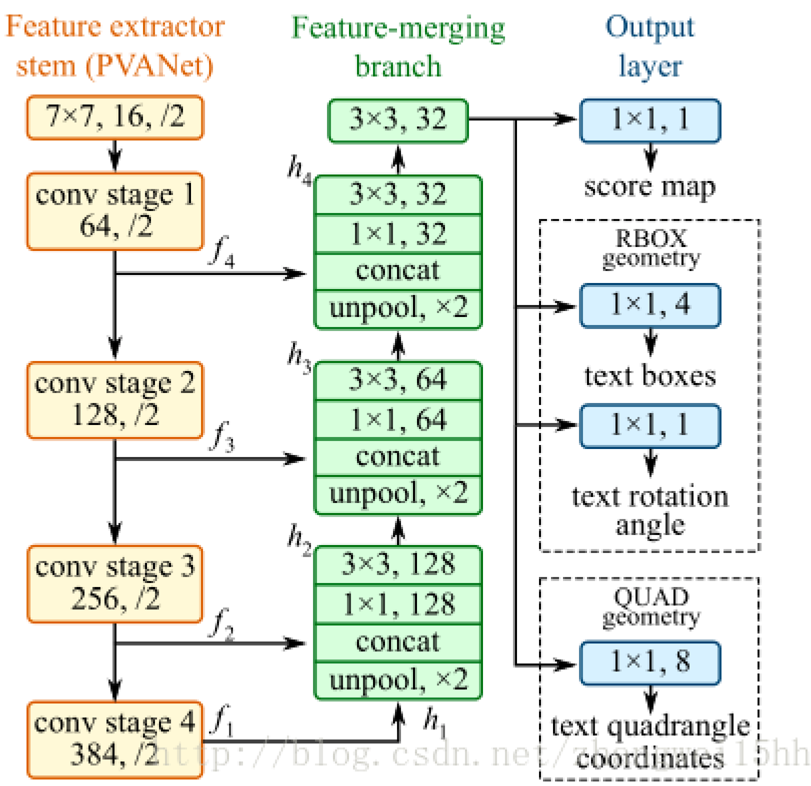


Figure 2. Comparison of pipelines of several recent works on scene text detection: (a) Horizontal word detection and recognition pipeline proposed by Jaderberg *et al.* [12]; (b) Multi-orient text detection pipeline proposed by Zhang *et al.* [48]; (c) Multi-orient text detection pipeline proposed by Yao *et al.* [41]; (d) Horizontal text detection using CTPN, proposed by Tian *et al.* [34]; (e) Our pipeline, which eliminates most intermediate steps, consists of only two stages and is much simpler than previous solutions.

该模型直接预测全图像中**任意方向**和四边形形状的单词或文本行，**消除不必要的中间步骤**（例如，候选聚合和单词分割）。通过下图它与一些其他方式的步骤对比，可以发现该模型的步骤比较简单，去除了中间一些复杂的步骤，所以符合它的特点，EAST, since it is an Efficient and Accuracy Scene Text detection pipeline.



网络结构

第一步

Feature extractor stem(PVANet)

利用Inception的思想，即不同尺寸的卷积核的组合可以适应多尺度目标的检测，作者在这里采用PVANet模型，我们也可以用VGG16或者其他的常见网络，**提取不同尺寸卷积核下的特征并用于后期的特征组合。**

第二步

Feature-merging branch

在这一部分用来组合特征，并**通过上池化和concat恢复到原图的尺寸**，这里借鉴的是U-net的思想。

所谓上池化一般是指**最大池化的逆过程**，实际上是不能实现的但是，可以通过只把池化过程中最大激活值所在的位置激活，其他位置设为0，完成上池化的近似过程。

$$g_i = \begin{cases} \mathrm{unpool}(h_i) & \text{if} \quad i \leq 3 \\ \mathrm{conv}_{3\times3}(h_i) & \text{if} \quad i = 4 \end{cases}$$

$$h_i = \begin{cases} f_i & \text{if} \quad i = 1 \\ \mathrm{conv}_{3\times3}(\mathrm{conv}_{1\times1}([g_{i-1}; f_i])) & \text{otherwise} \end{cases}$$


```
 for i in range(4):
        print('Shape of f_{} {}'.format(i, f[i].shape))
    g = [None, None, None, None]
    h = [None, None, None, None]
    num_outputs = [None, 128, 64, 32]
    for i in range(4):
        if i == 0:
            h[i] = f[i]  # 计算h
        else:
            c1_1 = slim.conv2d(tf.concat([g[i-1], f[i]], axis=-1), num_ou
tputs[i], 1)
            h[i] = slim.conv2d(c1_1, num_outputs[i], 3)
        if i <= 2:
            g[i] = unpool(h[i]) # 计算g
        else:
            g[i] = slim.conv2d(h[i], num_outputs[i], 3)
        print('Shape of h_{} {}, g_{} {}'.format(i, h[i].shape, i, g[i].s
hape))
```

第三步

Output Layer

第二部分的输出通过一个**（1x1，1）的卷积核**获得score_map。score_map与原图尺寸一致，每一个值代表此处**是否有文字的可能性**。

第二部分的输出通过一个**（1x1，4）的卷积核**获得RBOX 的geometry_map。有四个通道，分别代表**每个像素点到文本矩形框上，右，底，左边界的距离**。另外再通过一个（1x1, 1）的卷积核获得该框的旋转角，这是为了能够识别出有旋转的文字。

第二部分的输出通过一个（**1x1，8）的卷积核**获得QUAD的geometry_map，八个通道分别代表**每个像素点到任意四边形的四个顶点的距离。**

Geometry	channels	description
AABB	4	$\mathbf{G} = \mathbf{R} = \{d_i \mid i \in \{1,2,3,4\}\}$
RBOX	5	$\mathbf{G} = \{\mathbf{R}, \theta\}$
QUAD	8	$\mathbf{G} = \mathbf{Q} = \{(\Delta x_i, \Delta y_i) \mid i \in \{1,2,3,4\}\}$

代价函数

代价函数分两部分

- 第一部分是分类误差，
- 第二部分是几何误差，文中权衡重要性，λg=1。

$$L = L_s + \lambda_g L_g$$

具体参考blog

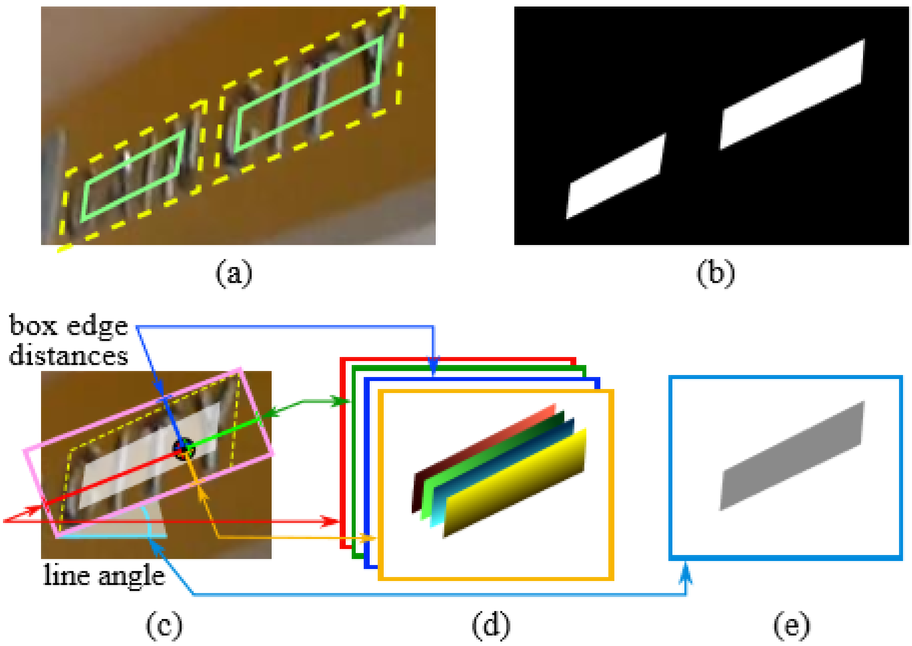


Figure 4. Label generation process: (a) Text quadrangle (yellow dashed) and the shrunk quadrangle (green solid); (b) Text score map; (c) RBOX geometry map generation; (d) 4 channels of distances of each pixel to rectangle boundaries; (e) Rotation angle.

分类误差函数

采用`class-balanced cross-entropy`，这样做可以很实用的处理正负样本不均衡的问题。来自Holistically-Nested Edge Detection

$$
\begin{aligned}
L_{\mathrm{s}} &= \text{balanced-xent}(\hat{\mathbf{Y}}, \mathbf{Y}^*) \\
&= -\beta \mathbf{Y}^* \log \hat{\mathbf{Y}} - (1-\beta)(1-\mathbf{Y}^*) \log(1-\hat{\mathbf{Y}})
\end{aligned}
$$

其中:

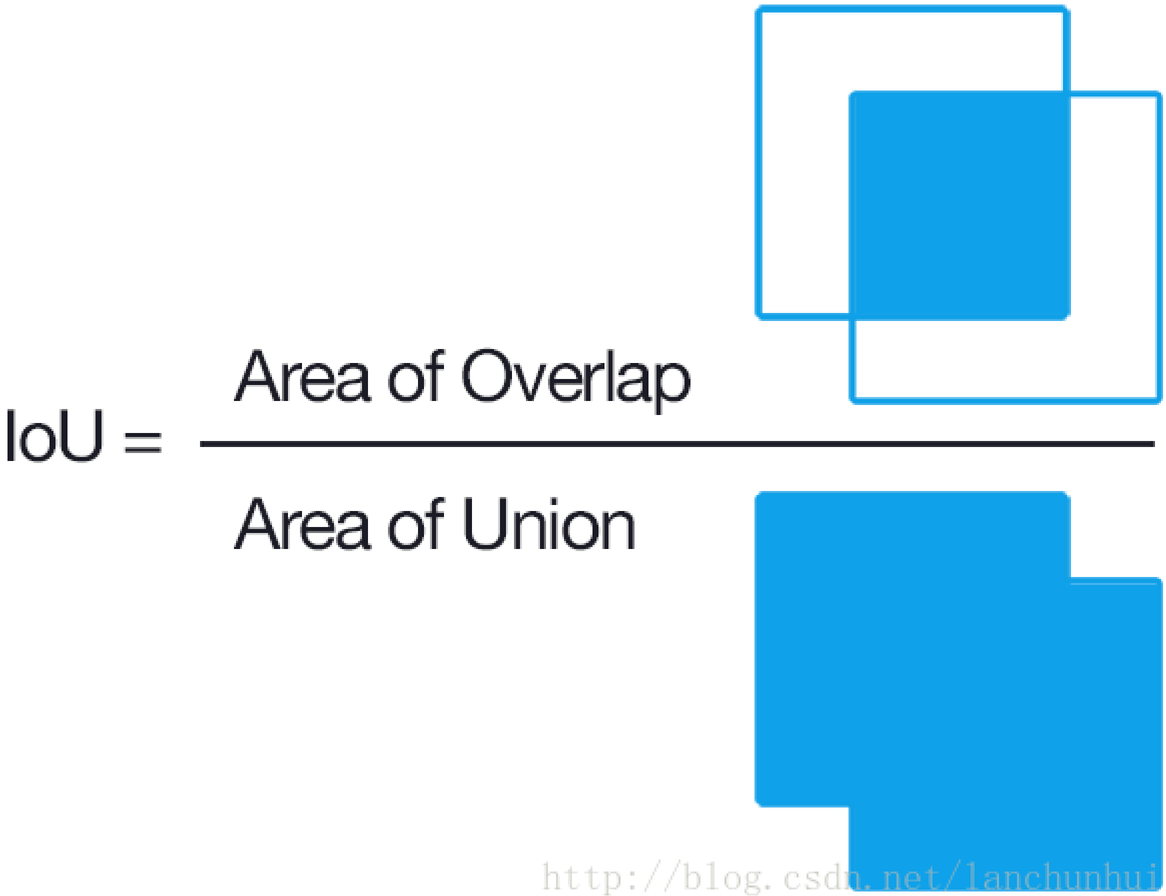
$$
\beta = 1 - \frac{\sum_{y^* \in \mathbf{Y}^*} y^*}{|\mathbf{Y}^*|}.
$$

几何误差函数

1、对于RBOX，采用IoU loss

$$
L_{\mathrm{AABB}} = -\log \mathrm{IoU}(\hat{\mathbf{R}}, \mathbf{R}^*) = -\log \frac{|\hat{\mathbf{R}} \cap \mathbf{R}^*|}{|\hat{\mathbf{R}} \cup \mathbf{R}^*|}
$$

where $\hat{R}$ represents the **predicted** AABB geometry and R^* is its corresponding ground truth.



角度误差为：

$$L_\theta(\hat{\theta}, \theta^*) = 1 - \cos(\hat{\theta} - \theta^*).$$

2、对于QUAD采用smoothed L1 loss

CQ={x1,y1,x2,y2,x3,y3,x4,y4}

$$\begin{aligned} L_{\mathrm{g}} &= L_{\mathrm{QUAD}}(\hat{\mathbf{Q}}, \mathbf{Q}^*) \\ &= \min_{\tilde{\mathbf{Q}} \in P_{\mathbf{Q}^*}} \sum_{\substack{c_i \in \mathrm{C}_{\mathbf{Q}}, \\ \tilde{c}_i \in \mathrm{C}_{\tilde{\mathbf{Q}}}}} \frac{\mathrm{smoothed}_{L1}(c_i - \tilde{c}_i)}{8 \times N_{\mathbf{Q}^*}} \end{aligned}$$

where $\hat{\theta}$ is the prediction to the **rotation angle** and θ^* represents the **ground truth**.

导入数据

```
icdar.py
        -generator:
            1、得到文本下的所有图片路径    image_list = np.array(get_images())
            2、循环读取图片和对应的txt文本信息内容
                  im = cv2.imread(im_fn)
```

```
                    text_polys, text_tags = load_annoataion(txt_fn)      #得到坐标和文本内容
                    text_polys:      文本坐标
                    text_tags :      内容信息
             3、 检测得到的文本坐标信息是否有效：
                    text_polys, text_tags = check_and_validate_polys(text_polys, text_tags, (h, w))
                    #检测得到的文本坐标是否有效
             4、按比例任意缩放图片大小
             5、从图片中任意剪切一块面积下来
                    im, text_polys, text_tags = crop_area(im, text_polys, text_tags, crop_background=True)
             6、最后经过一些列的计算得到
                    1、input_images: 图片
                    2、input_score_maps: 分数标记
                    3、input_geo_maps: RBOX
                    4、input_training_masks: 训练标记
```

具体流程参考load_image.py文件分析

评价指标计算

`python script.py -g='gt.zip' -s='submhome/weijia.wu/workspace/Paper/ICDAR_test/' -p='0.8'`

反馈与建议

- 微博：@柏林designer
- 邮箱：wwj123@zju.edu.cn