OCR-PixelLink 调研

2020 年 12 月 31 日

Contents

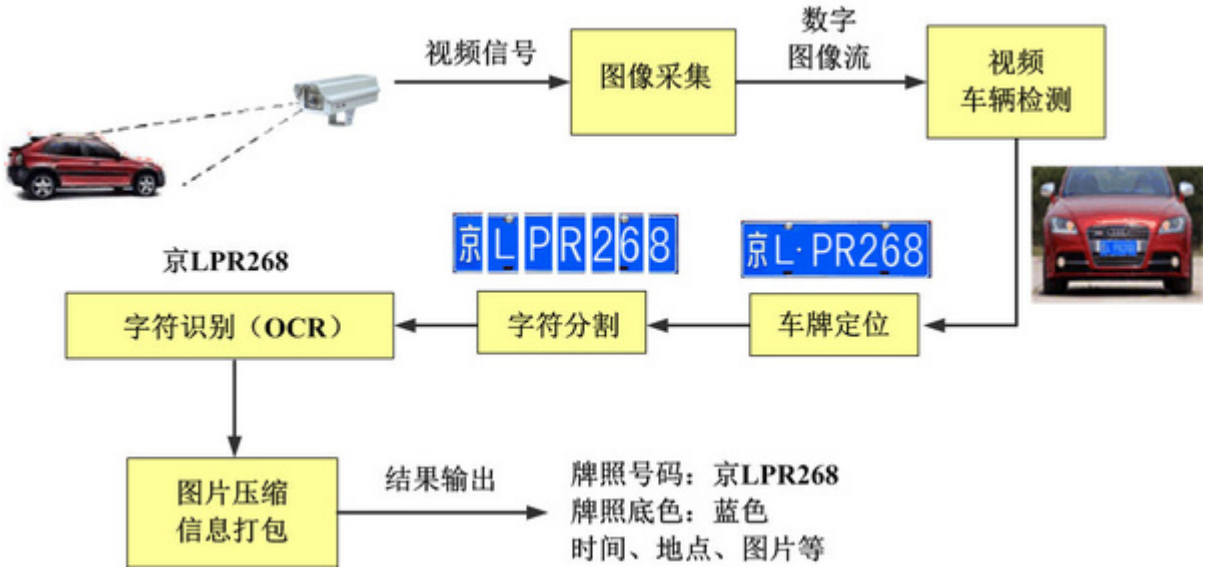
CHAPTER 1

应用综述

1.1 一. 基本原理:

光学字符识别（Optical Character Recognition，OCR）是指利用光学字符识别技术，对文本资料的图像文件进行分析识别处理，获取可编辑的文字及版面信息的过程。

车牌识别为例:



上图为车牌识别流程示意图

1. 预处理：噪声过滤、自动白平衡、自动曝光以及伽马校对、边缘增强、对比度调整等。
2. 车牌定位：对图像预处理后的灰度图像上进行扫描处理，判定车牌区域。
3. 字符切开：对图像进行灰度化道闸和二值化等处理，判定位字符区域，然后依据字符尺度特征进行字符切开。
4. 字符识别：对切开后的字符进行缩放、特征提取，与字符数据库模板中的标准字符表达形式进行匹配区别。

1.2 二. 应用场景：

特定场景识别：专用 OCR 识别功能，如证件识别、车牌识别、发票识别和银行卡识别。
通用场景识别：可用于更复杂的自然场景，在移动端也具有更大的应用潜力。

上图为自然场景中出现的字符识别

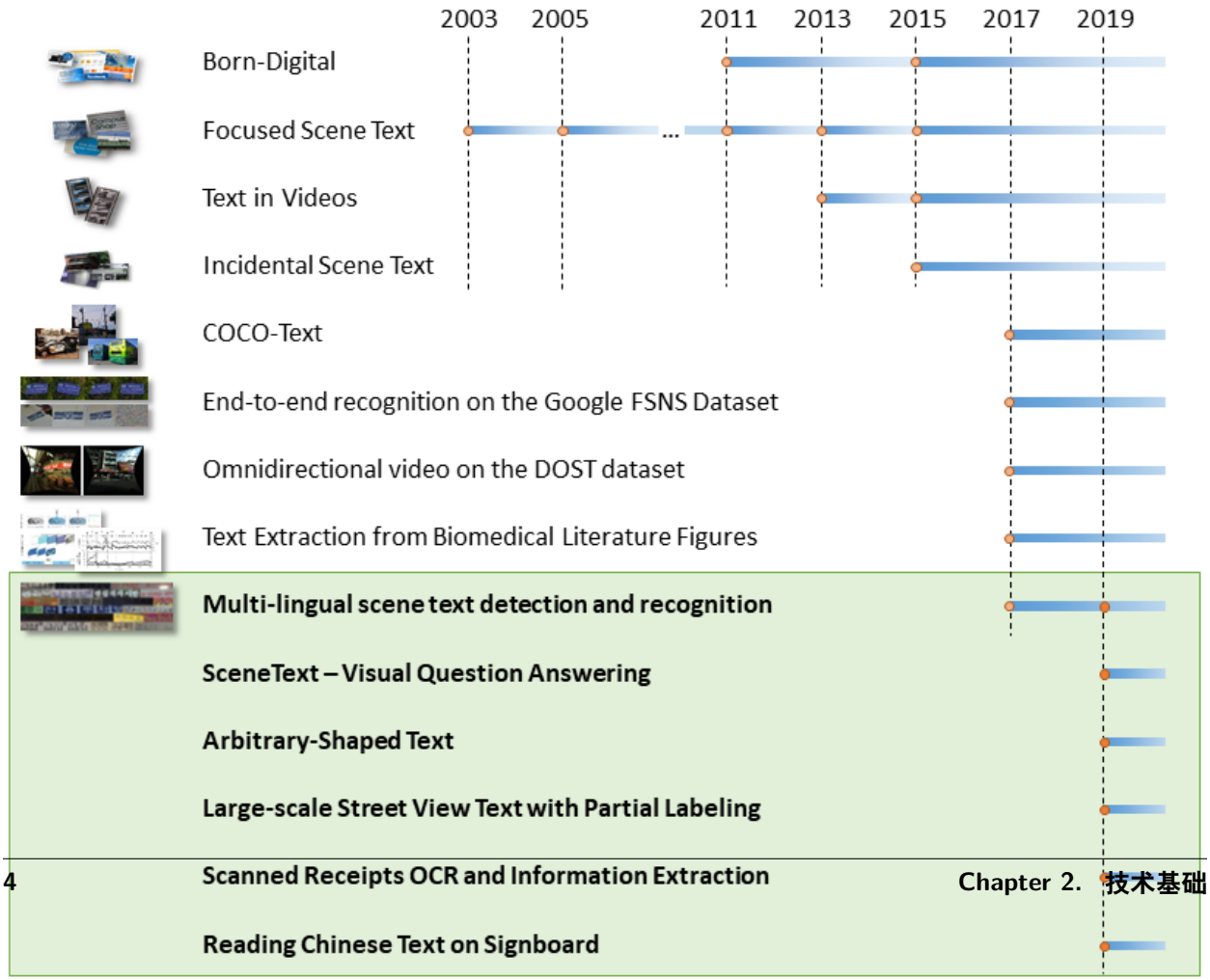
自然场景识别难点：

- 多种语言文本：文字可以有不同的大小、字体、颜色、亮度、对比度等。文本行可能有横向、竖向、弯曲、旋转、扭曲等式样。
- 文字形态复杂：文字可能出现产生变形 (透视、仿射变换)、残缺、模糊等现象。
- 背景多样：文字可能出现在平面、曲面或折皱面上等
- 同质干扰：文字附近有复杂的干扰纹理或者非文字区域有近似文字的纹理，比如沙地、草丛、栅栏、砖墙等。

CHAPTER 2

技术基础

2.1 一. 数据集:



- `ICDAR2015(IC15)`

IC15 的 Challenge 4 是 **检测任意方向场景文本**最常用的基准。它由两组组成: 训练和测试，分别包含 1000 和 500 张图像。图像是使用 Google Glass 获取的，不需要考虑视点、位置或帧质量。长度超过 3 个字符的可读拉丁语文字才会被标注为四边形。

- `ICDAR2013(IC13)`

IC13 是另一个被广泛使用的场景文本检测基准，包含训练图片 229 张，测试图片 233 张。这个数据集中的**文本实例大多是水平的，单词被标注为矩形**。

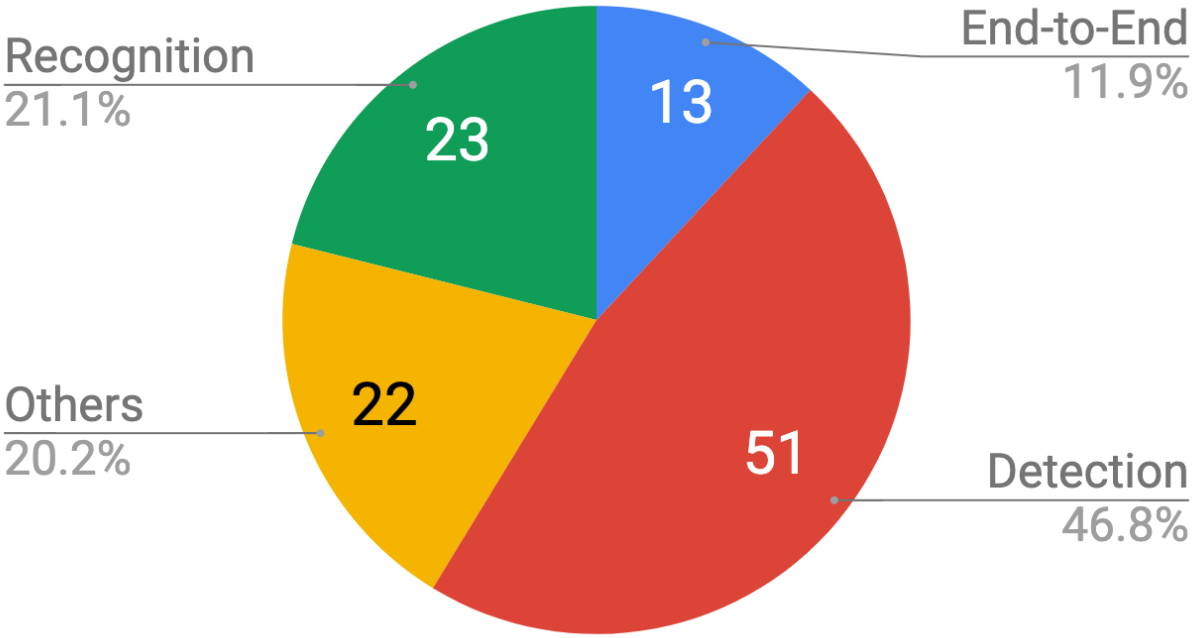
- `MSRA-TD500(TD500)`

TD500 中的文本也是任意方向的，但比 IC15 中的文本要长得多，因为它们是用 **行注释**的。TD500 总共包含 500 张图片，300 张用于培训，200 张用于测试。中英文都有。

2.2 二. 基础网络模块:

- 特征提取网络: 对图像中的文本进行特征提取，可来源于通用场景的图像分类模型。例如，**VGGNet，ResNet**、**InceptionNet** 等;
- 特定功能网络: 例如，适合提取图像细节特征的 **FCN 全卷积网络**，适合图像校正的 **STN 空间变换网络**等;
- 基础检测网络: 侧重检测精度的 **Faster-rcnn** 和侧重检测速度的 **SSD** 网络。

2.3 三. 基础模型:



2.3.1 1. 文本检测

1.1 评测方法:

OCR 竞赛 ICDAR 中将文本检测按照 **难度不同分为 Challenge1-4** 。其中 **图片中的文本定位**分为 Challenge 1、2 和 4，**视频中的图像定位**为 Challenge 3。主要介绍 Challenge 1、2 和 4 的异同:

- `Challenges 1` (Born-Digital) 的数据来源于 **电脑制作**;
- `Challenges 2` 主要是来源于用户有意识的 **对焦拍摄**的图像。比如一些翻译的场景，这些场景中文字基本是对焦好的且水平的;
- `Challenges 4` 主要来源也是用户拍摄，但是这些照片的拍摄是 **比较随意的场景**。通常情况下这些图片里的文字角度、清晰度、大小等情况十分复杂。

针对不同的挑战，有不同的评测方法; Challenges 1 和 2 使用的是 DetEva ； Challenges 4 通过 IoU 来判定算法的 recall、precision 的指标。

DetEval 方法

recallMat 和 **precisionMat** 中存储的是每个检测框的召回率和准确率，计算方法为:

$$\text{recall } Mat_{i,j} = \frac{\text{area}(\text{inter } (gt_i, \text{det}_j))}{\text{area}\,(gt_i)}$$

$$\text{precision } Mat_{i,j} = \frac{\text{area}\left(\text{inter}\left(gt_i, \det_j\right)\right)}{\text{area}\left(\det_j\right)}$$

其中 gt 表示标准框，det 表示检测框。area() 函数表示求矩形的 **面积**，inter() 函数表示求两个矩形的 **交集**。

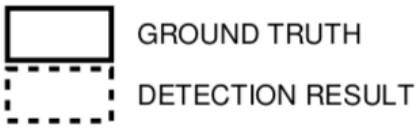
DetEval 方法考虑到三种情况，分别是图 a、b 和 c 中的三种。在考虑三种情况前，先进行以下定义：

1. 定义两个阈值 r 和 p ，r 表示判断召回率的阈值 0.8，p 表示判断准确率的阈值 0.4。
2. 定义 *RecallValue* 和 *PrecisionValue*，且初始值为 0。
3. 召回率 *Recall* 和准确率 *Precision* 分别为最终的 RecallValue 除以标准框数量和 PrecisionValue 除以检测框数量。

(a)

(b)

(c)



- `one-to-one matches (a)`：表示 **一个标准框对应一个检测框**。比较 recallMat 和 precisionMat 与阈值大小。判断是否为 one-to-one 的情况，若满足，就将 RecallValue 和 PrecisionValue 的数值加 **1**。
- `one-to-many matches (b)`：表示 **一个标准框对应多个检测框**。比较 recallMat 和 precisionMat 与阈值大小。判断是否为 one-to-many 的情况，若满足，则对于 RecallValue 值加上 **0.8**，PrecisionValue

值加上 **0.8** * many（对应检测框的数目）。

- `many-to-one matches (c)`：表示 **多个标准框对应一个检测框**。比较 recallMat 和 precisionMat 与阈值大小。判断是否为 many-to-one 的情况，若满足，则对于 RecallValue 值加上 **0.8** * many（对应检测框的数目），PrecisionValue 值加上 **0.8**。

最终评价指标：F1-Score，为 Recall 和 Precision 的 **调和平均数**。公式如下：

$$\mathrm{F1} = \frac{2 * P * R}{P + R}$$

IOU 方法

在 **iouMat** 中，统计大于 0.5 的个数，然后除以标准框的个数得到 Recall，除以检测框的个数得到 Precision。然后通过 **F1-Score** 方法得到最终评价指标 F1。

$$iouMat_{i,j} = \frac{\mathrm{area}(\mathrm{inter}\ \ (gt_i, \mathrm{det}_j))}{\mathrm{area}\left(\mathrm{union}\left(gt_i, det_j\right)\right)}$$

1.2 常用模型：

近年来出现了各种基于深度学习的技术解决方案。它们从 **特征提取、区域建议网络（RPN）、多目标协同训练、Loss 改进、非极大值抑制（NMS）、半监督学习**等角度对常规物体检测方法进行改造，极大提升了自然场景图像中文本检测的准确率。例如：

a. `CTPN`： 用 **BLSTM** 模块提取字符所在图像 **上下文特征**，以提高文本块识别精度。

b. `RRPN`： 文本框标注采用 **BBOX + 方向角度值**的形式，模型中产生出可旋转的文字区域候选框，并在 **边框回归计算**过程中找到待测文本行的倾斜角度。

c. `DMPNet`： 使用 **四边形（非矩形）标注文本框**，来更紧凑的包围文本区域。

d. `SegLink`： 将单词 **切割为更易检测的小文字块**，再 **预测邻近连接**将小文字块连成词。

e. `PixelLink`： 通过 **实例分割检测文本**，预测的 **正像素**通过预测的 **正链接**加入到文本实例中，然后直接从分割结果中提取边界框。

f. `EAST`：使用具有 **位置意识的 NMS（非最大值抑制）**进行非常密集的预测。

g. `TextBoxes`： 调整了文字区域参考框的 **长宽比例**，并将特征层 **卷积核调整为长方形**，从而更适合检测出细长型的文本行。

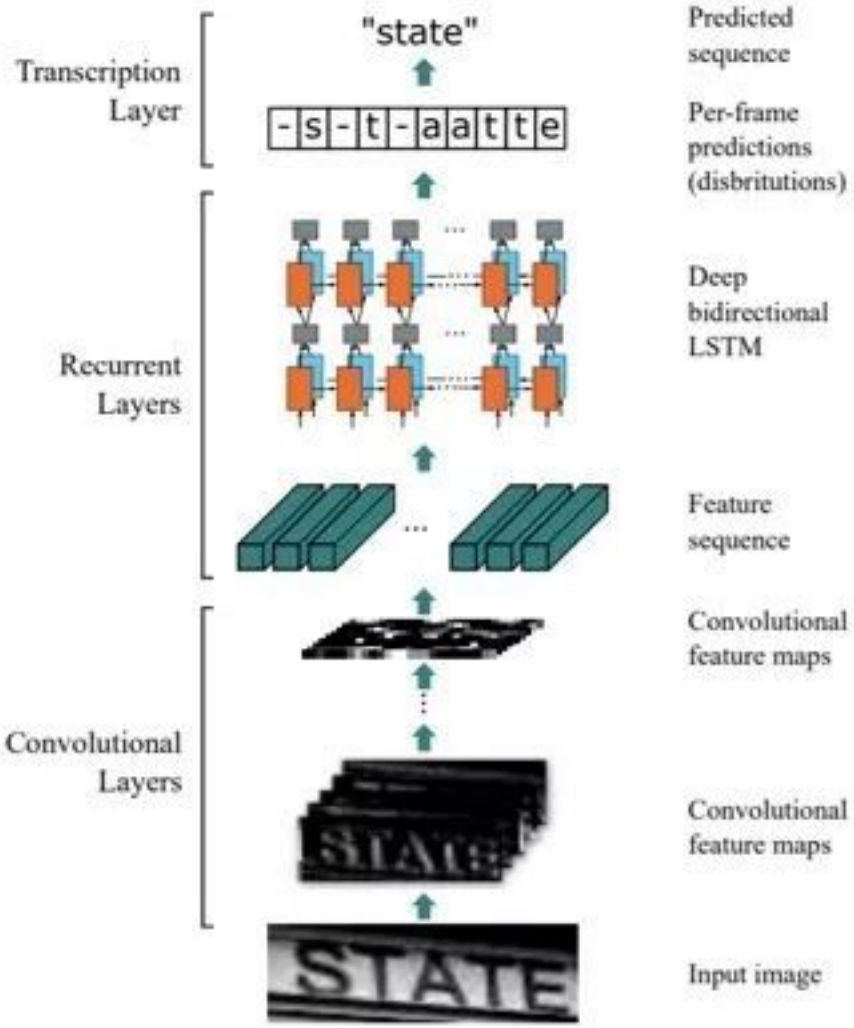
h. `FTSN`： 使用 **Mask-NMS** 代替传统 BBOX 的 NMS 算法来过滤候选框。

i. `WordSup`： 采用 **半监督学习策略**，用单词级标注数据来训练字符级文本检测模型。

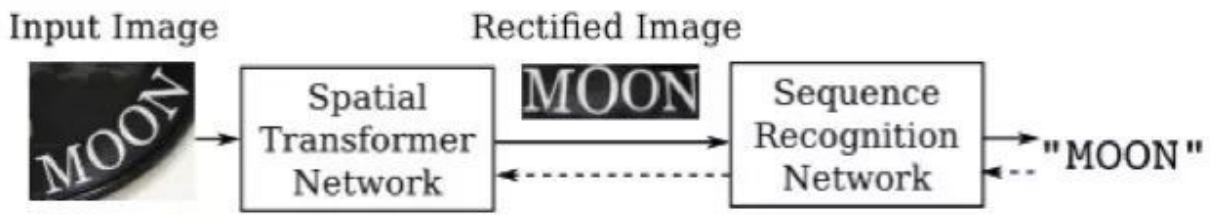
2.3.2 2. 文本识别

- `CRNN` 是目前较为流行的图文识别模型，可识别较长的文本序列。利用 **BLSTM** 将特征向量进行融合来 **提取字符序列的上下文特征**，然后得到每列特征的概率分布，最后通过 **转录层 (CTC rule)** 进行

预测得到文本序列。

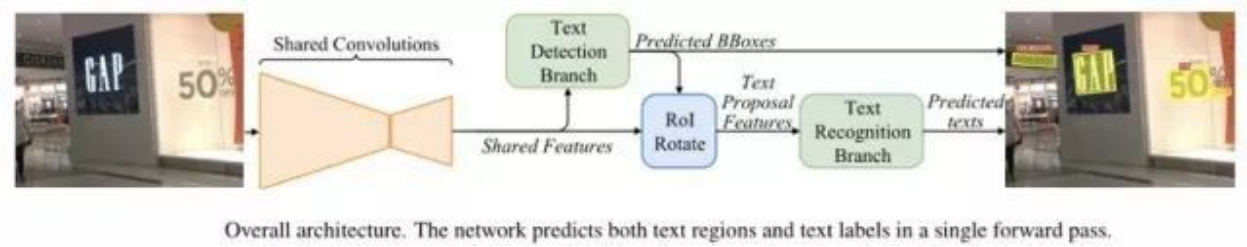


- RARE 模型在识别变形的图像文本时效果很好。模型预测过程中，输入图像首先要被送到一个 **空间变换网络**中做处理，矫正过的图像然后被送入 **序列识别网络**中得到文本预测结果。



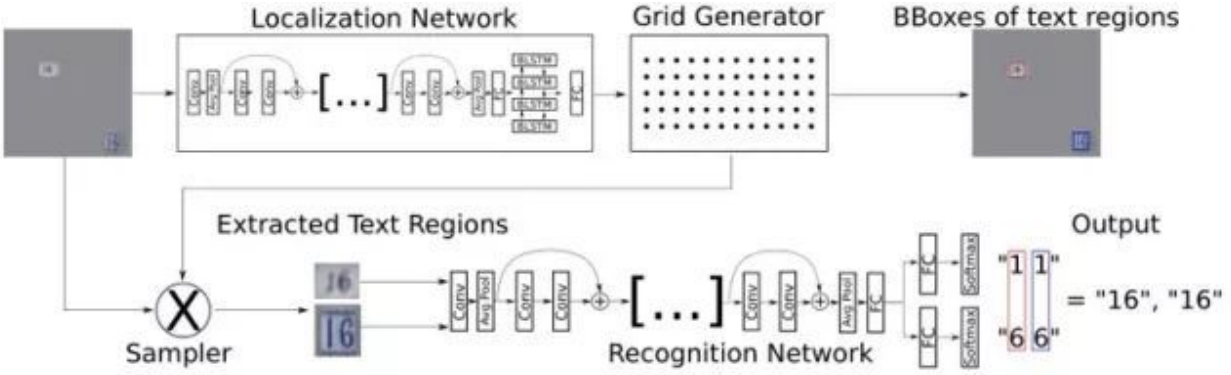
2.3.3 3. 端到端模型

- FOTS 是图像文本检测与识别同步训练、端到端可学习的网络模型。引入了 **旋转感兴趣区域 (RoIRotate)**，可以从卷积特征图中产生出定向的文本区域，从而 **支持倾斜文本**的识别。



Overall architecture. The network predicts both text regions and text labels in a single forward pass.

- STN-OCR 是集成了 **图文检测和识别**功能的端到端可学习模型。在它的检测部分嵌入了一个 **空间变换网络（STN）**来对原始输入图像进行仿射变换。利用这个空间变换网络，可以对检测到的多个文本块分别执行 **旋转**、**缩放和倾斜**等图形矫正动作，从而在后续文本识别阶段得到更好的识别精度。



注解： 相关网址：

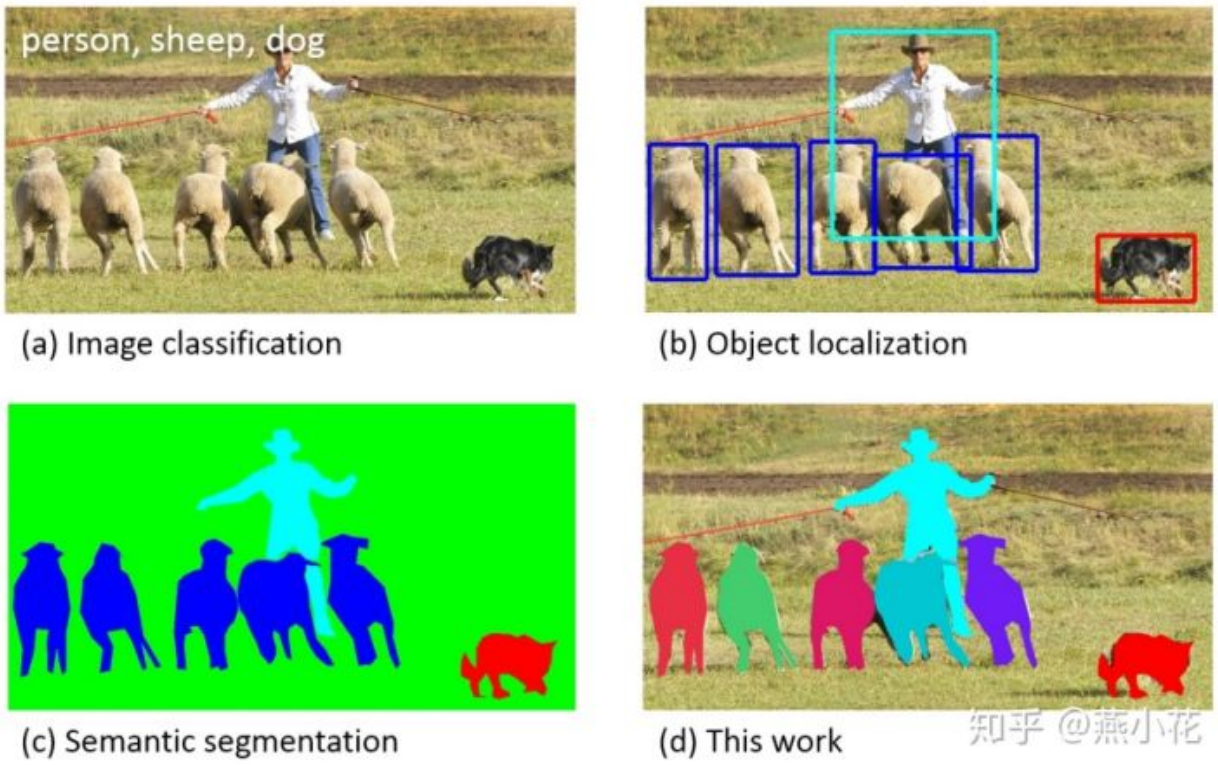
1. OCR 竞赛 ICDAR 网址：https://rrc.cvc.uab.es
2. 评测指标官方网址：https://rrc.cvc.uab.es/?com=faq
3. DetEval 方法：https://perso.liris.cnrs.fr/christian.wolf/software/deteval/inde
4. OCR 技术简介: https://zhuanlan.zhihu.com/p/45376274

CHAPTER 3

PixelLink 模型分析

首先将相同实例中的正像素通过正链接连在一起，然后来分割出来，最后直接从分割结果中提取文本边界框，不需要外置回归。

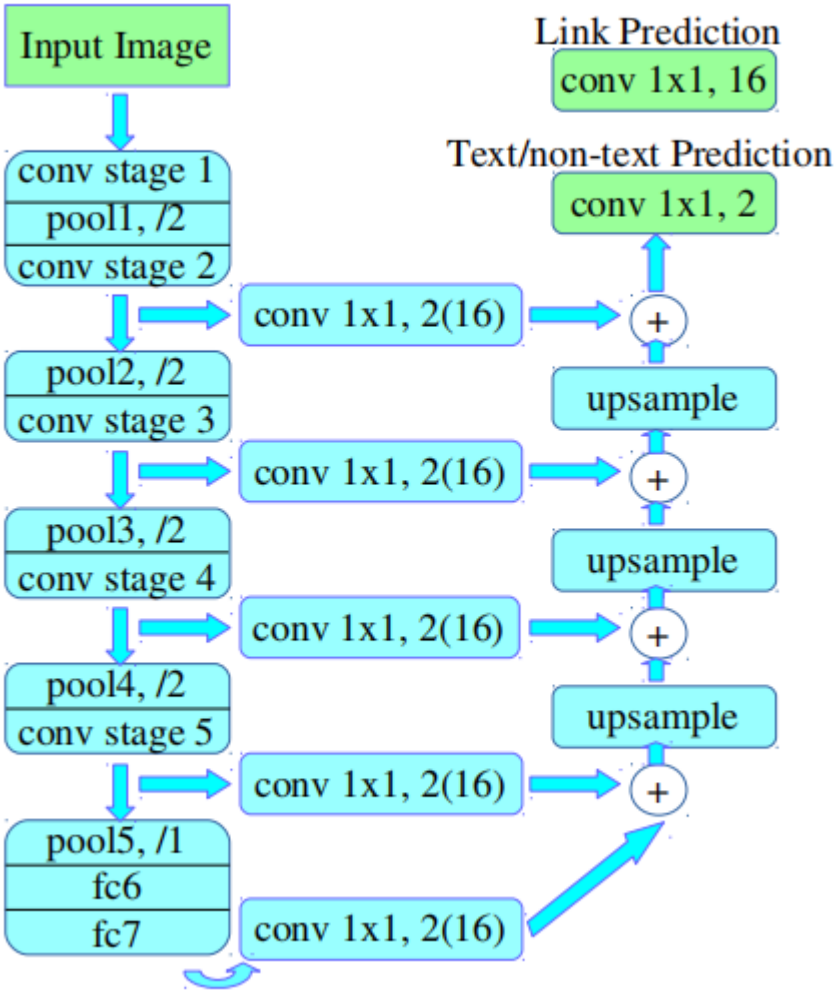
3.1 一. 基础工作:



上图分别是图像分类、目标定位、语义分割和实力分割的示意图。

实例分割是检测和分割每个目标实例；**语义分割**是为每个像素分配一个类标签。

3.2 二. 网络结构:



上图是 Pixel link 网络结构示意图

论文中给出了两种网络结构:

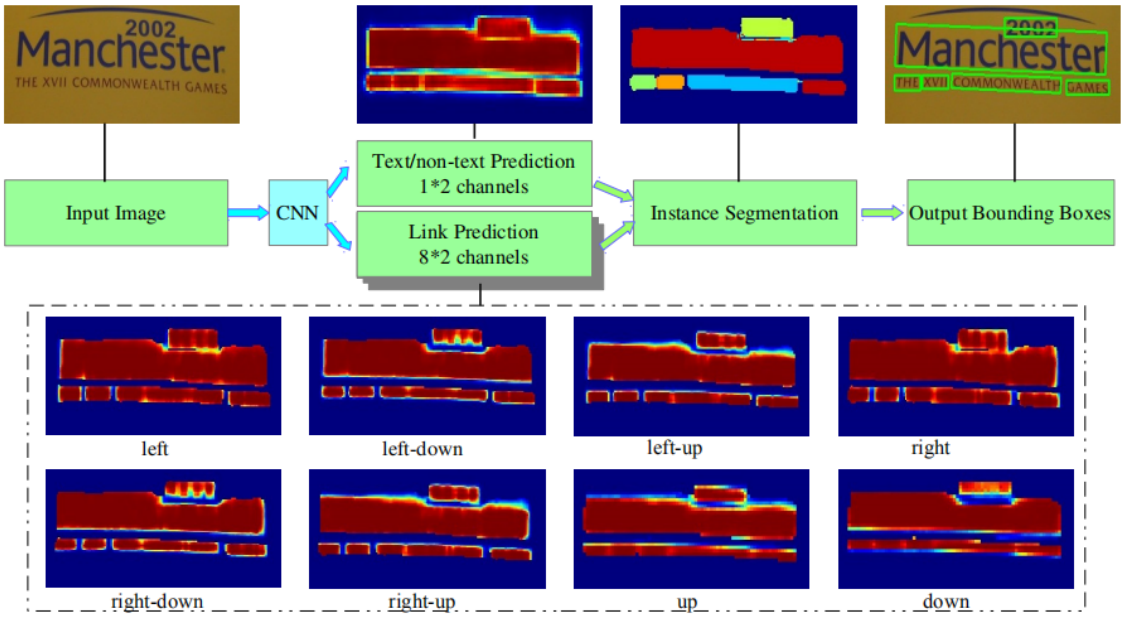
- `PixelLink+VGG16 2s` : 其融合的特征层包括, {conv2_2, conv3_3, conv4_3, conv5_3, fc_7}, 得到的特征图分辨率为原图的 **二分之一**;
- `PixelLink+VGG16 4s` : 其融合的特征层包括, {conv3_3,conv4_3, conv5_3, fc_7}, 得到的特征图分辨率为原图的 **四分之一**。

整个实现过程包括两部分: 先通过深度学习网络预测 **pixel positive** 和 **link positive**, 并根据 link positive 连接 pixel positive 得到 **文本实例分割图**, 然后从分割图中直接提取文本行的 bbox。具体步骤如下:

1. 对输入图片进行特征提取: 主干网络沿用了 **SSD** 网络结构, 以 **VGG16** 作为基础网络, 并将 VGG16 的最后两个全连接层改成 **卷积层**;

2. 提取不同层的特征图：对于 **PixelLink+VGG16 2s** 网络结构：提取了 conv2_2, conv3_3, conv4_3, conv5_3, fc_7；
3. 对提取后的特征层采用自顶向下的方法进行融合：融合操作包括先向 **上采样**，然后再进行 **add 操作**；
4. 获得网络输出进行实例分割：包括 **文本/非文本预测和 Link 预测**，即为 **pixel positive** 和 **link positive**，分别对应输出的通道数为 2（1*2）和 16（8*2）。

3.3 三. 重点实现：

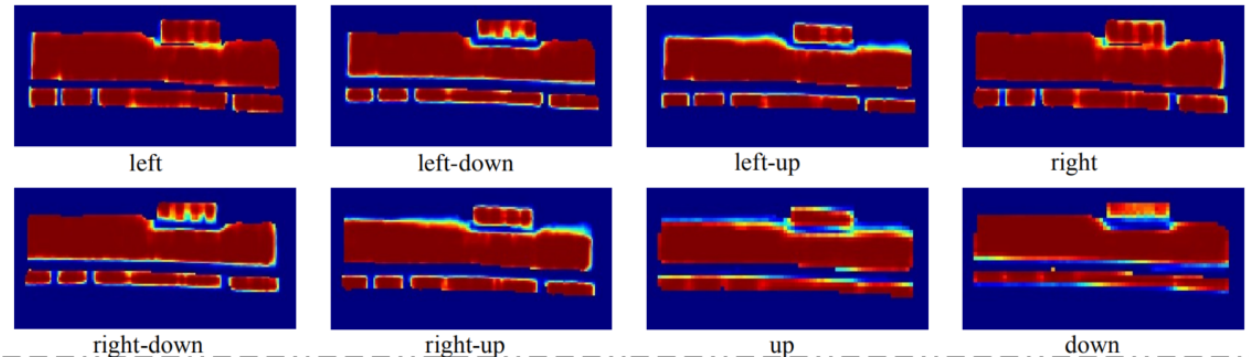


上图是 Pixel Link 模型结构示意图

具体实现步骤如下：

1. 训练 CNN 模型进行 **文本/非文本预测和链接预测**。
2. 经过阈值处理后，用 **并查集**方式，将正像素通过正链接连接在一起，实现实例分割。
3. 应用 **MinAreaRect** 直接从分割结果中提取边界框。
4. 后处理通过 **噪声滤波**筛选需要的文本框

3.3.1 1. 连接像素 (Linking Pixels Together)



上图是 PixelLink 体系结构中链路预测的热力图，代表 8 个方向的链路预测。

连接像素的步骤如下：

1. 设定阈值（大于 0.5），得到 pixel positive 集合和 link positive 集合。
2. 根据 link positive 将 pixel positive 进行连接，得到 CCs (conected components) 集合。
3. 根据集合得出分割的实例块，集合中的每个元素代表的就是文本实例。

1	2	3/1	2	3
4	Pixel A	5/4	Pixel B	5
6	7	8/6	7	8

上图是像素 A 和 B 分别在 8 个方向 link 链接的示意图

连接规则：

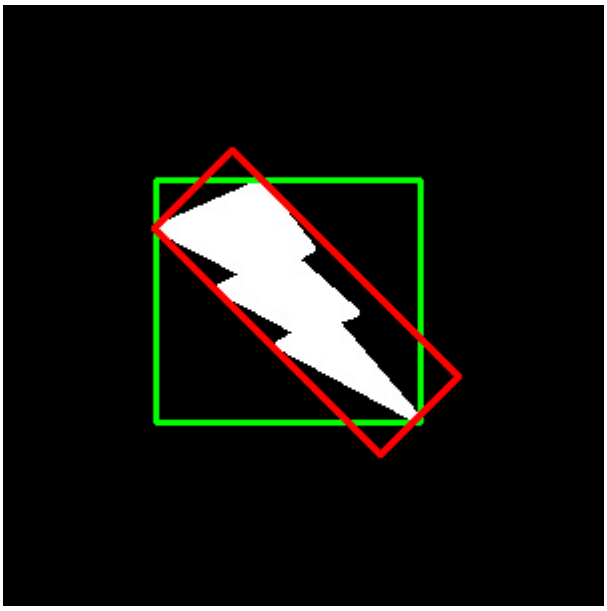
- 给定两个相邻的 pixel positive，它们之间的 link 预测是由当前 **两个 pixel 共同决定**的。如上图所示，像素 A 和 B 之间的链接 5 和 4 是由像素 A 和 B **共同决定**的。
- 两个 pixel 连接 (即两个像素属于同一个文本实例) 的前提条件：**two link 中至少有一个 link positive**。如上图所示，连接像素 A 和 B 的前提是链接 5 和 4 中 **至少有一个为 link positive**。

对于在文本/非文本预测中很难区分的文本，通过链接预测可以区分开。

3.3.2 2. 提取文本行的 bounding box（Extraction of Bounding Boxes）

提取文本框步骤：基于上述分割的结果，直接通过 opencv 的 MinAreaRext 提取文本的带方向信息的外接矩形框 (即带角度信息)。矩形框的格式为 ((x,y),(w,h),θ)，分别表示 **中心点坐标，当前 bbox 的宽和高，旋转角度**。

MinAreaRext 方法



上图表示的是 MinAreaRect 方法，从图中可以看出此方法在得到的是图形的最小外接四边形。

PxielLink 和 SegLink 的区别在于：PixelLink 是直接从分割结果中提取 bbox，而 SegLink 采用的是边框回归。

3.3.3 3. 实例分割的后处理（Post Filtering after Segmentation）

- 进行后处理的原因是在 **pixel 进行连接的时候会引入噪声**。
- 通过统计训练集中样本的情况，得到 **95% 概率以上的共有特征**，根据此几何特征制定筛选规则。
- 利用筛选规则对已检测到的 bbox 进行判断筛选，最终提高 **文本检测**的准确率。

3.4 四. 训练技巧:

总损失函数

$$L = \lambda L_{pixel} + L_{link}$$

3.4.1 1. 实例平衡交叉熵损失（Instance-Balanced Cross-Entropy Loss）

平衡 **不同大小的文本检测框**对损失函数的影响。

$$w_i = \frac{B_t}{S_t}$$

$$B_i = \frac{S}{N}, S = \sum_{i}^{N} S_i, \forall i \in \{1, \ldots, N\}$$

公式中 S 代表所有文本框的面积之和, B 表示平均每个文本框的面积。

3.4.2 2. 在线困难样本挖掘（Online Hard Example Mining）

平衡 **正负样本数量**不同对于训练过程中分类效果的影响。

$$L_{pixel} = \frac{1}{(1+r)S} W L_{pixel\_CE}$$

注解： 相关网址：

1. 文本检测之 PixelLink：https://zhuanlan.zhihu.com/p/38171172
2. PixelLink 论文地址：https://arxiv.org/pdf/1801.01315.pdf

CHAPTER 4

总结

4.1 一. PixelLink 网络调研:

步骤	项目	网络实例
前期调研	网络名称	PixelLink
	功能描述	OCR领域中的文本检测环节
	论文地址	https://arxiv.org/pdf/1801.01315.pdf
	源码地址	https://github.com/ZJULearning/pixel_link
	源码框架	tensorflow
	源码语言	python
	相关论文	https://arxiv.org/pdf/1703.06520.pdf
	评价方法	DetEva、IOU、F-measure
	原理解析	https://zhuanlan.zhihu.com/p/38171172
	可视化工具	netron、tensorboard
	数据集	MSRA-TD500、ICDAR2013、ICDAR2015
	待补充	
前处理	归一化处理	减均值：123(R), 117(G), 104(B)
	rgb通道的转换	rgb
	摆数策略	nchw -> nhwc
	resize尺寸变换	双线性resize： （768，1280）
	待补充	
推理	输入数据个数及规模	输入仅为图片：(BS, 768, 1280, 3)
	输出数据个数及规模	两个；一个是文本/非文本的分类得分（BS, 192, 320); 第二个是每个像素四周的链接的得分 (BS, 192, 320, 8)
	首层算子类型	首层为卷积层，backbone为 VGGnet，网络框架类似SSD。
	在线逐层	暂无调研
	生成离线模型	StrideSlice（具体代码：self.link_pos_scores = self.pixel_link_scores[:, :, :, :, 1])
	不支持op调研	[1,192,320,8,2]，cnml不支持大于四维的，可以用其他方式替代
	精度和性能数据	暂无调研
	待补充	
后处理	滤波	去掉检测框的短边小于10或者面积小于300的检测文本框 （10和300是IC15训练数据的统计结果）
	生成指定格式文件	生成 zip 文件，包含有框的信息。可以使用工具用于DetEva方法的评测
	待补充	
后期总结	调研输出	
	待补充	

上图是 PixelLink 网络调研表格输出，对应的网址请看页尾的注释。

4.1.1 1. 移植细节：

- 网络的源码是从论文提供的官方 github 中获取，同时可移植性测试是在 **MLU100，V8.2.1** 软件栈中进行。
- 目前不支持算子为 **StrideSlice** 。主要在网络的最后一层，其作用是对 **五维 tensor 进行切分**，得到所需要的四维 tensor。根据源码分析，可通过 **修改算子实现**解决此问题。
- 目前没有确切的网络移植需求以及客户的具体网络实现，因此还 **未进行精度和性能的评测**。

4.1.2 2. 移植分析：

- 文字检测：PixelLink 是文字检测领域中比较常用的一个网络框架。其中 backbone 为 **SSD 网络**，而 MLU 对 SSD 已有适配，因此 PixelLink 对 MLU 来说 **可移植性强**。
- 文字识别：当前通用的文字时别算法采用了 **RNN 回归网络**，而 MLU 之前也有成功的 RNN 网络的落地（百度语音项目）。

综上所述，MLU 产品适合进入 OCR 领域，进行相关网络的适配工作。同时根据已有的经验和调研分析，可以提高适配后网络的性能同时保证精度。

4.2 二. 当前挑战：

首先，它与传统 OCR（光学字符识别）有所区别，因自然场景之中文字变化非常之多，如下图所示：左边是一张典型的 **扫描文档图像**，右边则是 **自然场景**中采集的多张图像。

通过对比可以发现:

- 左图的背景非常 **干净**，右图的背景非常 **杂乱**；
- 左边字体非常 **规整**，右边则 **千变万化**；
- 左边布局较为 **平整统一**，右边则 **多元复杂，缺乏规范**；
- 左边 **颜色单调**，右边 **颜色种类繁多**。

目前来看，深度学习时代之下的场景文字检测与识别技术依然存在巨大挑战，其主要体现在以下 **3 个方面**：

文字的差异性：存在着语言、字体、方向、排列等各种各样的形态；

背景的复杂性：比如几乎无法区分的元素（标志、篱笆、墙砖、草地等）；

干扰的多样性：诸如噪声、模糊、失真、低分辨率、光照不均匀、部分遮挡等情况。

4.3 二. 未来机遇：

相应而言，针对上述挑战，文字检测与识别具体研究有着以下 **4 个技术趋势**：

1. 设计更强大的模型；
2. 识别多方向、弯曲文字；
3. 识别多语言文字；

4. 合成更丰富逼真的数据集。

注解： PixelLink 网络调研相关：

1. 论文地址：https://arxiv.org/pdf/1801.01315.pdf
2. 源码地址：https://github.com/ZJULearning/pixel_link
3. 相关论文：https://arxiv.org/pdf/1703.06520.pdf
4. 原理解析：https://zhuanlan.zhihu.com/p/38171172

相关网址：

1. 深度学习时代的文字检测与识别技术：https://zhuanlan.zhihu.com/p/51725259
