代码和数据集在https://github.com/FudanVI/benchmarking-chinese-text-recognition

# Introduction

本文试图构建一个中文文本识别的基准来填补这一空白。首先从公开竞赛、论文和课题中提取已有的中文文本数据集，得到四个类别，包括场景、网页、文档和手写体（请参见图2）。然后，以合理的比例将每个数据集手动划分为训练、验证和测试集。请注意，与现有的英语文本基准不同，特别设计验证集来公平地比较现有的方法，即确保最佳模型是基于验证集而不是测试集选择的。此外，还再现了一些具有代表性的文本识别方法(如CRNN[59]，aster[60]，moran[45]等)的结果。在收集的数据集中作为基准的基准线。通过对基线实验结果的分析，我们发现一些最初为英语文本设计的最先进的方法并不能很好地适应中文文本。通过深入的分析，发现中文文本的一些特征（如长文本、大字母表、复杂的语义特征）确实以不同的方式对现有的方法设置了障碍。还证明了数据集构建的合理性，在每个类别的可识别性，并由人类校准。总的来说，本文的贡献可以列为如下：

1. 从公共竞赛、论文和项目中手动收集中文文本识别数据集。然后，将它们分为四类，包括**场景、网络、文档和手写数据集**。进一步以合理的比例将这些数据集划分为训练、评估和测试集
2. 提出了一些评价方法来公平地比较现有的文本识别方法
3. 基于收集到的数据集和所提出的评价方法，再现了一系列基线的结果，然后详细分析了基线的性能

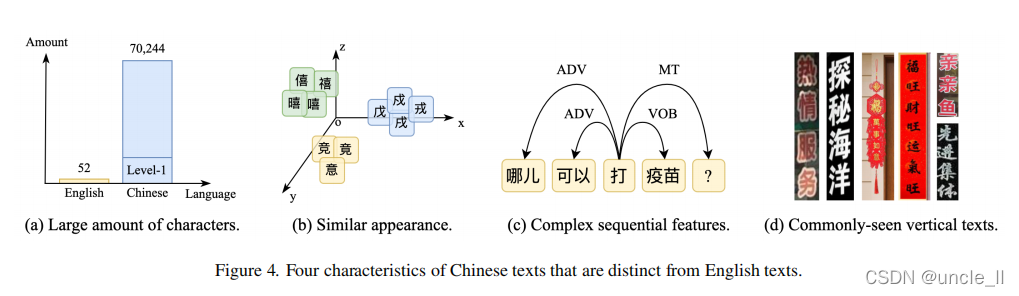


# 准备工作Preliminaries

在本节中，讨论汉语文本与英语文本不同的一些特征。然后介绍了现有的文本识别方法。

## 中文文本的特点

* 大量的字符；
* 相似性；
* 复杂的顺序特征；
* 常见的竖排文本；



## 已有的文本识别方法

* 自上而下
* 自下而上

其中，自上而下的方法将每个单词视为唯一的类别，并执行词级分类，而自下而上的方法通过执行字符级识别，然后将它们分组为单词，将每个字符视为类别。

Jaderberg等人[31]提出了一种具有代表性的自顶向下方法，利用文本图像的词级深层特征对9万个常用词进行词级分类。早期的自下而上的方法，如PhotoOCR[7]和Tasseract[62]，倾向于使用图像处理操作定位精确文本区域，并使用SVM或其他线性分类器对分割的区域执行字符识别，最后将分离的字符片段分组为一个完整的单词。

随着深度学习的蓬勃发展，研究者们也仿照自下而上的方法，尝试建立基于深度神经网络的文本识别模型例如，CRNN[59]使用CNN-RNN体系结构来提取文本图像的特征，并在CTC损失[24]的情况下进一步监督这些特征，以最大化标签真实的概率。然而，由于CRNN在解码时将特征的高度限制为1，因此在自然场景中容易受到一些失真的文本图像的影响。在这种情况下，提出了一些基于纠错的方法来缓解这一问题。例如，ASTER[60]利用SpatialTransformer Networks[32]通过控制点来纠正扭曲的文本。Moran[45]首先以弱监督的方式生成像素方向的偏移图，然后通过像素采样对文本图像进行校正。除了以上方法外，其他方法尝试在2D特征域中进行文本识别[36，37]。例如，SAR[36]直接利用二维特征进行解码。[37]利用字符级标注，利用二维特征映射对每个字符的分割掩码进行预测。此外，一些著作[21，58，82]也试图在自然语言处理领域蓬勃发展的Transformer[64]的基础上构建极端认知器，通过自我注意模块来稳健地学习文本图像的表示。近年来，一些研究者将语义知识融入到文本识别器中，以充分利用外部语言先验知识[20，54，75，88]。例如，SEED[54]利用由FastText[8]引导的文本嵌入来初始化基于注意力的解码器。Abinet[20]设计了两个自治分支来迭代优化视觉和语言模型。总体而言，这些通用的文本识别方法可以很容易地转移到中文场景中，只需简单地替换字母或语言优先级。然而，如果进一步考虑到中文文本的特点，它们的表现仍有很大的提高空间。

# 数据集Datasets

在这一部分中，首先介绍中文文本识别数据集的预处理方法，然后详细介绍每个类别(如场景、Web、文档和手写)的数据集。最后，对收集到的数据集进行了分析。

## 准备

* 保留包含其他语言字符的样本。在日常生活中，会见到各种语言的字符，例如中文、英文、日文、韩文等，因此为了追求实用性，依然保留数据集中其他语言的样本。
* 移除标签为“###”的样本。一般情况下，数据集标注者会将肉眼难以鉴别的样本标注为“###”。因此将这部分样本从数据集中移除，避免这些样本给识别模型带来困扰。
* 为每个数据集制作验证集。观察到大多数方法直接在测试集上筛选最优模型，这样并不严谨，因此我们强调了验证集的重要性，并为每个数据集划分出一部分作为验证集。
* 不使用没有标签的样本。许多竞赛数据集的测试集并没有label，不使用这一部分数据集。

基于这些原则，从可用的资源中收集了四类数据集，下面将介绍这四类数据集。

## 数据详情

### 场景数据|Scene dataset

从竞赛、论文和项目中收集了RCTW [61], ReCTS [96], LSVT [63], ArT [17], CTW [89]这几类数据

* RCTW：该数据集提供来自自然场景的12,263个带注释的中文文本图像。对于该数据集，从训练集提取44,420行文本行，并在基准测试中使用它们。由于文本标签不可用，因此不使用RCTW的测试集
* ReCTS：该数据集提供了25,000个带注释的街景中文文本图像，这些图像主要来自路面的招牌。对于该数据集只采用训练集，总共裁剪了107,657个文本样本作为基准
* LSVT：该数据集是一个大规模的中英文场景文本数据集，提供5万个全标注(多边形框和文本标签)和40万个部分标注(每幅图像只有一个文本实例)样本。对于该数据集，只使用全标签训练集和裁剪243,063个文本行图像作为基准
* ArT：该数据集包含在自然场景中捕获的各种文本布局(如旋转文本和弯曲文本)的文本样本，从训练集中获得49,951幅裁剪文本图像，并将它们用于基准测试
* CTW：该数据集包含30,000个带注释的街景图像，具有丰富的多样性，包括平面、凸起和弱光文本图像。此外，它不仅提供字符框和标签，还提供诸如背景复杂性、外观等字符属性。对于该数据集，从训练和测试集中裁剪191,364行文本

组合所有子数据集，得到636,455个文本样本。随机洗牌这些样本，并按8：1：1的比例拆分，得到509,164个样本用于训练，63,645个样本用于验证，63,646个样本用于测试。

### web数据|web dataset

为了收集网络数据集，利用MTWI[26]，它包含来自淘宝网站17个不同类别的20,000个中英文网络文本图像。文本样本出现在各种场景、字体和设计中，从训练集中提取了140589幅文本图像，并按8：1：1的比例进行人工分割，得到112,471个样本用于训练，14,059个样本用于验证，14,059个样本用于测试。

### 文件数据集|Document dataset

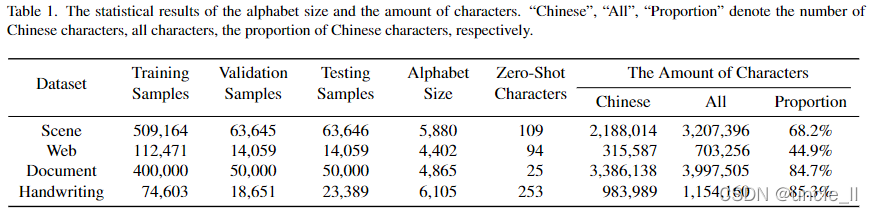
使用公共库Text\_Render( https://github.com/Sanster/text\_renderer)生成一些文档式的合成文本图像，更具体地说，对长度从1到15的文本进行统一采样。语料库来自维基、电影、亚马逊和百科。数据集总共包含500,000个，并且被随机分为训练、验证和测试集，比例为8：1：1(400,000 : 50,000 : 50,000)。

### 手写数据集|handwriting dataset

本文收集了基于SCUT-HCCDoc[93]的手写数据集，该数据集在不受约束的环境中使用相机捕获中文手写图像。根据官方设置，分别获得了93,254个用于训练的文本行和23,389个用于测试的文本行。为了进行更严格的研究，将原始训练集按4：1的比例手动分割为两个集，得到74,603个样本用于训练，18,651个样本用于验证。为方便起见，继续使用原来的23,389个样本进行测试。

## 样本数据分析

### 分析字母表大小和字符数



从上图可以看到，字母表的大小和汉字的比例在四个数据集中都不同。例如，大多数网络文本都是为世界各地的客户提供固定表达的中文广告，因此字母表中包含的字符较少(4402个字符)。此外，网络数据集包含了大量的电话号码和英文网站，导致汉字在四个数据集中的比例最低(44.9%)。对于手写数据集，大多数文本都是从中国古诗中抄袭过来的，与其他数据集相比，中国古诗中包含了更多的不常用字符，因此产生了最大的字母表(6105个字符)。此外，手写数据集的测试集包含了训练集中缺失的最多的zero-shot字符(253个)，这进一步增加了识别的难度。

### 文本长度和长宽比的分布分析

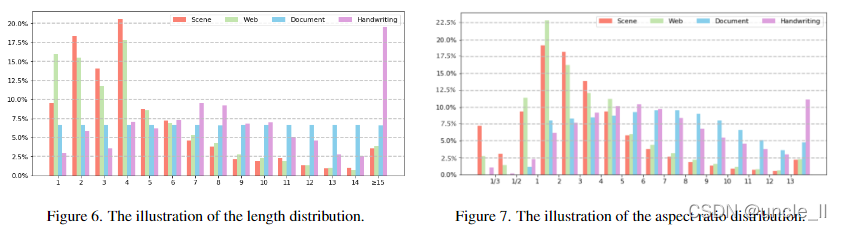
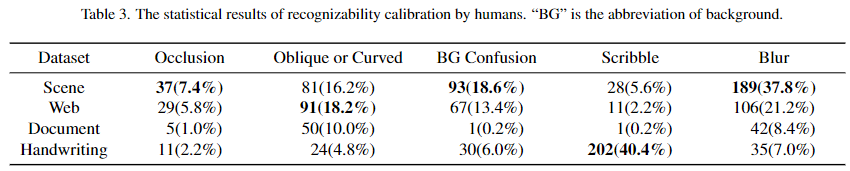


图6和图7说明了文本长度和纵横比(即宽高比)的分布。从这些图中，观察到长文本(例如，Length≥10)更频繁地出现在手写数据集中，这给基线带来了困难。相反，场景和网页数据集中的文本相对较短，可能是考虑到乘客、顾客等的阅读效率。在比例分布方面，观察到由于野外常用的对联和招牌，场景数据集的垂直文本（ratio<=1）比其他数据集多。相比之下，手写数据集包含更多的水平文本。

### 由人类进行可识别性校准



为了找出影响文本图像在四个数据集中的可识别性的因素，邀请了20名受过高等教育的参与者进行了这项实验。邀请参赛者找出阻碍识别的相应原因(多项选择)：

(1)遮挡(被其他物体遮挡)

(2)倾斜或弯曲(文字为倾斜或弯曲)

(3)背景模糊(读者很难将前景字符和背景纹理分开)；

(4)涂鸦(经常出现在手写文字中，有些笔迹可能会加入或省略笔划，以方便阅读)；

(5)模糊(低焦距相机拍摄的图像)。

如果这些原因都不能满足，则认为文本图像是可识别的。

对每个数据集都有500票(每个数据集采样25个图像样本，20个参与者)。统计数据如表3所示。可以看到

* 在场景数据集中，参与者投票的原因最多的是\*\*“遮挡”、“背景混淆”和“模糊”\*\*，这表明这是人类能够识别的最复杂的数据集。影响识别率的因素主要来自自然环境和文字图像的采集方式。
* 对于web数据集，通过可控的生成，使文本外观更加灵活随意，从而产生了“倾斜或弯曲”选择的最多票数。虽然曲线文字是为了更好的视觉效果而设计的，但严重扭曲的字符有时不容易让人识别。
* 对于文档数据集，样本对人类来说相对容易识别，因此投票率较低。
* 对于手写数据集，参与者投票最多的是“涂鸦”，说明书写中的笔画衔接也是影响识别性的一个基本因素。通过以上观察，可以发现影响识别率的因素在不同的数据集上各不相同，从而促使我们分别研究每个数据集的性能

# 基线Baselines

文本识别在过去的十年中取得了快速的进步。根据主要特征，文本识别方法可以分为几类，包括基于CTC的方法、基于纠错的方法、基于Transformer的方法等。从这些类别中，选择了七种有代表性的方法作为基线(具体的特征如表4所示)，下面将对这些方法进行介绍。

* CRNN：该算法首先将文本图像送入CNN进行图像特征提取，然后采用两层LSTM对序列特征进行编码，最后将LSTM的输出反馈给CTC(Connectionist Temperal Class)[24]解码器，最大化所有路径朝向真实的概率。
* ASTER：该算法是一种典型的针对不规则文本图像的基于校正的方法。引入了空间变换网络(STN)[32]来将给定的文本图像校正为更易识别的外观。然后将经过认证的文本图像分别送入CNN和两层LSTM进行特征提取。特别地，ASTER利用注意力机制来预测最终的文本序列。
* MORAN：该算法是一种有代表性的修正为主的方法。该算法首先采用多目标校正网络(MORN)以弱监督的方式预测校正后的像素偏移量(不同于利用STN的ASTER)，然后利用输出的像素偏移量生成校正后的图像，并将其送入基于注意力的解码器(ASRN)进行文本识别。
* SAR：该算法是一种利用二维特征映射实现更稳健解码的代表性方法，特别是主要针对撞击不规则文本提出了一种新的方法。一方面，SAR在CNN编码器中采用了更强大的残差块[25]来学习更强的图像表示。另一方面，与CRNN、ASTER和MORAN将给定的图像压缩成一维特征图不同，SAR采用二维关注特征图的空间维度进行解码，在弯曲和倾斜文本中具有更强的性能。
* SEED：该算法是一种典型的基于语义的方法。引入一个语义模块来提取全局语义嵌入，并利用它来初始化解码器的第一个隐藏状态。具体地说，SEED的解码器在继承ASTER结构的同时，引入语义嵌入，为识别过程提供先验信息，在识别低质量文本图像时显示出优越性。
* SRN：该算法是一种典型的基于语义的方法，利用自我注意模块来校正预测错误。提出了并行视觉注意模块和自注意网络，通过多路并行传输捕捉全局语义特征，显著提高了不规则文本识别的性能。
* TransOCR：该算法是基于transformer的方法中具有代表性的方法之一。它最初的设计目的是为超分辨率任务提供文本优先选项。它采用ResNet-34[25]作为编码器，自关注模块作为解码器。与基于RNN的解码器不同，自关注模块能够更有效地捕捉给定文本图像的语义特征。

## 实验研究

首先介绍了实现细节和评估方法，然后展示实验结果。最后对实验结果进行了详细的分析，并给出了每个数据集中的一些失败案例。

### 实现细节

采用现成的pytorch方法：

CRNN: https://github.com/meijieru/crnn.pytorch

ASTER: https://github.com/ayumiymk/aster.pytorch

MORAN: https://github.com/Canjie-Luo/MORAN\_v2

SEED: https://github.com/Pay20Y/SEED

SRN: https://github.com/PaddlePaddle/PaddleOCR

SAR: https://github.com/liuch37/sar-pytorch

TransOCR: https://github.com/FudanVI/benchmarking-chinese-text-recognition/tree/main/model

根据经验将场景、网页、文档、手写数据集的输入大小限制为64×200、64×200、64×800和64×1200。

利用每个数据集的验证集来根据识别精度确定最佳模型，然后使用测试集对基线进行评估。为了方便起见，将所有实验的四个数据集的字母表结合在一起，得到了7934个字符的总字符表。基线没有使用其他策略，如数据增强、模型集成、预训练。

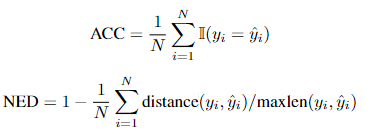
### 评估方法

在实践中，为了公平比较基线，统一的评价方法是必不可少的。根据ICDAR2019 ReCTS竞赛，提出了一些规则来转换预测和标签：

* 将全角字符转换成半角字符
* 将繁体汉字转换为简体字符
* 将大写字母转换为小写字母
* 删除所有空格

使用两个主流评价标准：

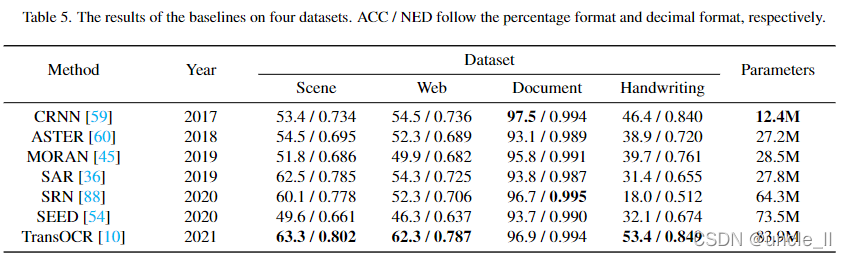
* 准确度(ACC)
* 归一化编辑距离(NED)



N是文本图像的数量，ACC和NED都在[0，1]范围内。ACC和NED越高，表明评估基线的性能越好。

### 实验结果分析

在四个数据集上对基线的性能进行了全面的分析。实验结果如表5所示：



* 场景数据

如表3所示，由于遮挡、背景混淆和模糊等问题，场景数据集相当具有挑战性。此外，如图7所示，场景数据集包括许多垂直文本图像，这确实给那些简单地将原始输入转换为一维特征序列的基线(例如，CRNN[59]、ASTER[60]和Moran[45])设置了障碍。相比之下，二维方法(例如，SAR[36]，SRN[88]和TransOCR[10])在这个数据集上取得了更好的性能，因为二维特征地图对于具有特殊布局(例如，垂直或弯曲)的撞击文本图像更具鲁棒性。此外，TransOCR[10]通过利用自注意模块，由于它能够更加灵活地对序列特征进行建模，因此其识别率为63.3%。最后，注意到SEED[54]在这个数据集上表现得不是很好，推测SEED需要在fast Text[8]的指导下将每个文本图像映射到其对应的语义嵌入，而中文文本通常包含复杂的语义特征，这给语义学习过程带来了困难

* web数据

web数据集比场景数据集相对容易区分。但注意到，所有基线的性能都低于场景数据集中的性能，这可能是由于训练样本的稀缺(用于web场景训练样本的个数为112,471个，而用于场景的样本为509,164个)。此外，注意到CRNN[59]、ASTER[60]、SAR[36]、SRN[88]的性能确实表现出很大的差异(即在53附近波动)。同时，垂直文本图像的比例(参见图7)比场景数据集的比例小，因此选择一维或二维特征地图不会导致本质上的差异。最后，由于TransOCR[10]强大的自注意模块，在识别准确率和归一化编辑距离方面也取得了最好的性能。

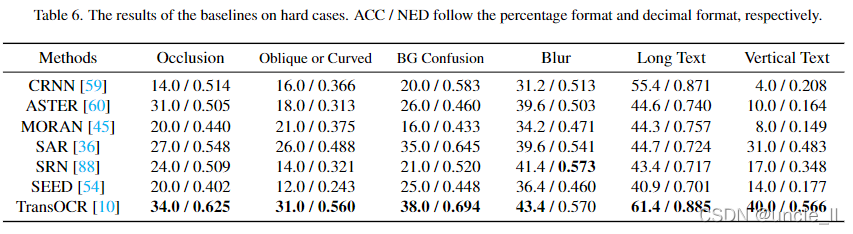
* 文档数据

由于文档数据集是使用文本呈现来合成的，因此与场景和Web数据集相比，此数据集中的文本样本更容易识别。\*\*此外，根据表3，几乎没有人将文本实例分类为遮挡、涂鸦或模糊。\*\*观察到所有基线的识别准确率都可以超过90.0%，值得一提的是，基于CTC的CRNN[59]在该数据集中的识别准确率最高，因为该数据集中没有垂直文本图像。\*\*换句话说，对于这样一个简单的场景，二维特征图和校正策略并不是必需的。此外，由于存在许多具有长文本(即，文本长度大于10)的图像，因此基于注意力的识别器(例如，ASTER[60]、Moran[45]和SEED[54])由于可能遭受注意力漂移问题而低于平均水平。

* 手写体数据

如表3所示，手写数据集中几乎40%的文本样本标记为“涂鸦”。在实践中，作者可能会为了加快书写速度而拼接或省略一些笔画，这确实给现有的方法带来了困难。通过实验结果，观察到这七个方法性能差异很大。例如，与基于校正的ASTER[60]和Moran[45]相比，CRNN[59]在这个数据集中取得了更好的性能，因为大多数文本样本都是水平的，并且序列很长。相反，基于语义的SRN[88]和SEED[54]的性能低于平均水平。注意到，在这个数据集中有很多不常见的语料库，包括唐诗和宋词，这可能会给SRN和SEED获取语义特征带来障碍。最后，TransOCR[10]获得了最好的识别效果，识别率为53.4%，进一步验证了自注意模块的强大能力。

### 困难样例分析



手动选择了一些困难的情况(例如遮挡、倾斜或弯曲、背景混淆、模糊、过长和垂直)，并分析了基线在这些情况下的性能。请注意，“涂鸦”的情况主要与手写数据集有关，因此不会单独分析“涂鸦”的情况。表6显示了在这些困难样本下的实验结果，可以看到，与其他基线相比，TransOCR[10]能够很好地适应每个困难情况。得益于自注意模块，识别器可以灵活地关联地图中的每个特征对来增强序列表示，从而使得很容易解决不常见的文本布局，例如倾斜、弯曲等，或者缓解前景(遮挡)或背景混乱带来的噪声。有趣的是，与Transocr[10]相比，SRN[88]在模糊情况下获得了更好的NED，因为它可以充分利用语义线索来弥补丢失的细节。此外，基于CTC的CRNN[59]在处理涉及NED的长文本方面表现出优势，因为它可以缓解基于注意力的解码器引起的注意力漂移问题。对于垂直文本图像，注意到使用二维注意力图如SAR[36]、SRN[88]和TransOCR[10]的基线比其他基线表现得更好。总体而言，在这些困难的样本下，还有很大的改善空间。

### 失败样例可视化

分别为场景、Web、文档和手写数据集可视化了图8、图9、图10和图11中的一些失败案例。



对于场景数据，注意到遮挡的文本(例如，“纽恩泰”，“财经天下”，“冰淇淋鸡蛋仔”和“海”)确实给识别器带来了困难，因为前景可能被错误地理解为文本的一部分。此外，还有一些非常困难的情况，如镜像文本(例如，“麦当劳”)，即使是人眼也看不清。



对于Web数据集，注意到基线很难处理艺术字体的文本图像(例如，“遇见”，“魅力端午”，“没有地沟油”和“我爱姓名贴”)。



对于文档数据集，如图10所示，尽管文档数据集具有最高的可识别性，因为它是通过文本呈现合成的，但是一些样本仍然对所有基线造成困难。基线没能处理一些很少使用的字符(如“鼾”，“轭”和“谯”)，从而错误地将它们视为其他相似的字符。



对于手写数据集，注意到连接或缺失的笔画可能会混淆基线(例如，“欢迎”和“火夏”)。此外，一些文本较长的图像可能会使基线很难捕捉到序列特征。