

数字图像处理课程设计报告

——车票序列号检测与识别

张启哲 1900011638

2022 年 1 月 13 日

目录

1 任务介绍	1
2 运行说明	1
3 算法原理	1
3.1 票面检测	1
3.1.1 二值化与去噪	1
3.1.2 形态学处理	1
3.1.3 票面定位与转正	2
3.1.4 票面裁剪与放缩	2
3.2 21 位码定位与分割	3
3.2.1 21 位码粗定位	3
3.2.2 21 位码精定位	3
3.2.3 21 位码分割	3
3.3 7 位码定位与分割	4
3.3.1 7 位码定位	4
3.3.2 7 位码分割	5
3.4 数字与字母识别	5
3.4.1 数据集建立	5
3.4.2 模型结构	5
3.4.3 预测后处理	6
4 实验结果	6
4.1 定位与分割结果	6
4.2 预测结果	6
5 分析讨论	6
6 总结	7
7 致谢	7

1 任务介绍

本次课程设计的任务是给定火车票的扫描图片, 对车票票面进行检测, 并对序列号进行定位与分割, 最后输出 7 位码与 21 位码的识别结果.

车票上的序列号分为 7 位码与 21 位码, 7 位码与 21 位码的后 7 位相同, 都是第 1 位为大写字母, 后 6 位为数字, 而 21 位码的前 14 位全部为数字. 任务给出 100 张带标注的火车票图片作为训练数据, 每张图片均有 7 位码与 21 位码的注释, 要求对 100 张测试图片进行检测与识别.

2 运行说明

预训练模型存储在 models 目录下, 测试文件为 predict.py, 测试时需要将测试集图片放在 test_data 目录下, 包含测试图片名从文件命名为 annotation.txt, 可以直接运行测试文件, 也可以指定 `-input_dir`, `-annotation_file`, `-output_dir`, `-prediction_file`, 这些参数的默认值都与要求一致, 预测结果中的检测与分割结果存储在 segments 目录下, 序列号识别结果写入 prediction.txt 文件中. 代码已上传至 github.com/Theia-4869/Railway-Ticket-Identification.

3 算法原理

3.1 票面检测

3.1.1 二值化与去噪

首先, 读取原始灰度图片 (Fig.1), 以 127 为阈值对原始图片进行二值化 (Fig.2), 由于背景噪声的存在, 二值化后背景区域可能出现线状白色条纹, 采用大小为 7×7 的中值滤波处理后 (Fig.3), 背景噪声条纹基本消失.

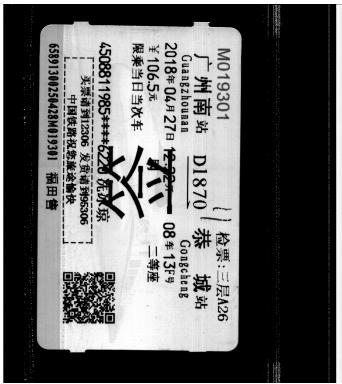


Figure 1: 原始灰度图片



Figure 2: 二值化后的图片



Figure 3: 中值滤波后的图片

3.1.2 形态学处理

为了得到更精确的票面区域, 在中值滤波后采用形态学处理来去除票面上多余的文字与线条等图案, 以及车票两个宽边可能出现的用于连接的小矩形部分. 首先, 用 7×7 的矩形结构元进行开操作来修整车票两个宽边的小矩形 (Fig.4), 之后用 70×70 的矩形结构元进行闭操作来去除票面上的文字与线条等 (Fig.5), 最后再用 70×70 的矩形结构元进行开操作去除车票两侧的突出小矩形 (Fig.6), 得到近似为矩形的干净的票面部分.



Figure 4: 第 1 次开操作后

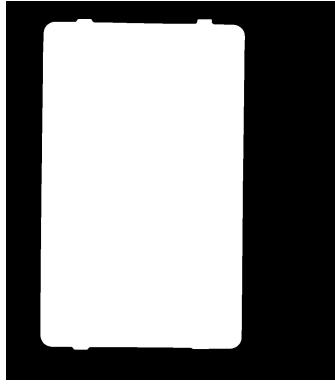


Figure 5: 闭操作后

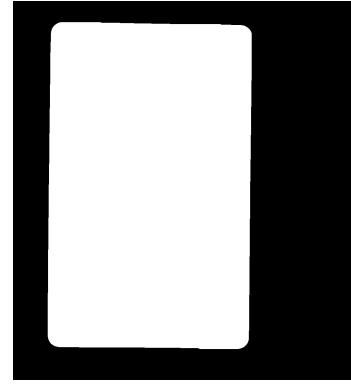


Figure 6: 第 2 次开操作后

3.1.3 票面定位与转正

形态学处理后, 利用 numpy 的 column_stack() 与 opencv 的 minAreaRect() 函数得到票面区域的最小矩形包围盒, 并用 drawContours() 将其画在票面上. 再利用得到的矩形包围盒估计旋转角度, 使用 warpAffine() 函数对图片进行旋转, 此时得到的票面仍有可能是上下颠倒的 (Fig.7), 再根据图片上半部分与下半部分的灰度值的比较, 判断是否需要将其旋转 180° 得到最终的转正后的票面 (Fig.8).



Figure 7: 根据包围盒角度转正后



Figure 8: 根据上下区域灰度修正后

3.1.4 票面裁剪与放缩

得到了转正的有包围盒的票面后, 根据包围盒裁剪出只包含票面的区域, 再统一缩放为 1080×640 的大小 (Fig.9) 方便后续的序列号定位.



Figure 9: 裁剪与放缩后的票面部分

3.2 21 位码定位与分割

3.2.1 21 位码粗定位

首先, 使用大小为 5×5 高斯模糊核与 30 的二值化阈值对票面部分进行处理, 再利用掩膜获取票面左下角包含 21 位码的一部分区域作为粗定位的操作区域 (Fig.10). 之后, 使用定位票面时的方法获取当前票面部分的最小矩形包围盒, 以其左下角作为定位点, 采用固定长宽对 21 位码区域进行粗定位 (Fig.11).

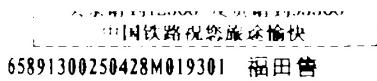


Figure 10: 粗定位操作区域

65891300250428M019301

Figure 11: 粗定位结果

3.2.2 21 位码精定位

随后, 根据粗定位得到的区域重复上述操作一次, 得到更精细的矩形包围盒, 该矩形包围盒即为 21 位码区域的精定位 (Fig.12), 将其绘制在图片上.



Figure 12: 精定位结果

3.2.3 21 位码分割

利用得到的包围盒作掩膜提取 21 位码区域 (Fig.13), 再通过 opencv 的 findContours() 函数对每个连通域单独计算最小矩形包围盒, 在每个包围盒的左边界处划分即得到 21 位码的分割结果 (Fig.14) (为了确保分割过程在相邻码位之间有粘连时的鲁棒性, 程序检查前一步是否得到 21 个连通区域, 若不是则重新采用等距划分).

65891300250428M019301



Figure 13: 21 位码区域



Figure 14: 21 位码分割结果

3.3 7 位码定位与分割

3.3.1 7 位码定位

由于 7 位码的灰度值与其他印刷字体不同,首先采用双阈值 (50:140) 对票面部分进行二值化处理 (Fig.15),再利用形态学处理 (2×2 的矩形结构元腐蚀接续 9×9 的矩形结构元闭操作)大致消除票面上的其余印刷字体轮廓 (Fig.16).

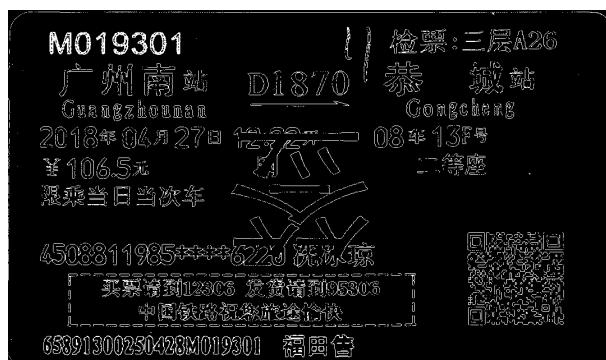


Figure 15: 7 位码双阈值二值化

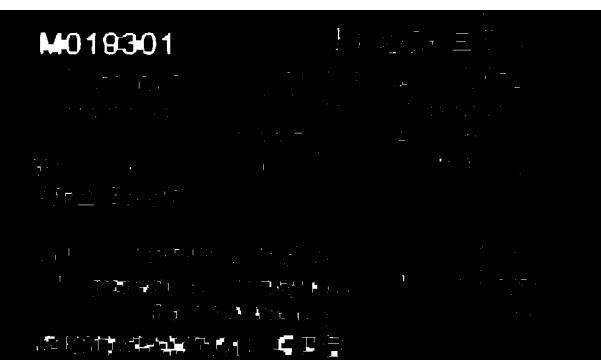


Figure 16: 7 位码形态学处理

形态学处理后,利用掩膜获取票面左上角固定的一部分区域作为 7 位码区域的大致位置,再去除连通域大小 <150 的噪点得到只含 7 位码的区域 (Fig.17),计算其最小矩形包围盒并绘制在图片上即得到 7 位码区域的定位结果 (Fig.18).

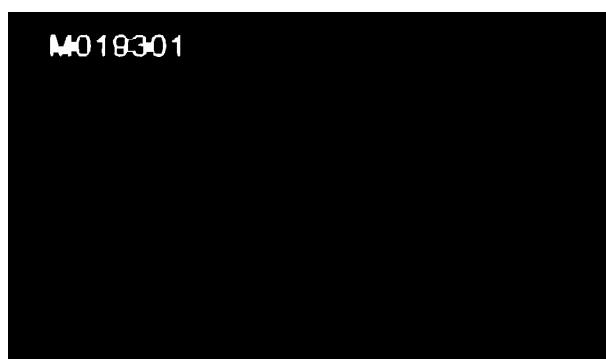


Figure 17: 7 位码粗定位结果



Figure 18: 7 位码精定位结果

3.3.2 7 位码分割

与 21 位码的分割相同, 也是利用得到的包围盒作掩膜提取 7 位码区域 (Fig.19), 再对每个连通域单独计算最小矩形包围盒, 在每个包围盒的左边界处划分即得到 7 位码的分割结果 (Fig.20) (同样为了确保分割过程在相邻码位之间有粘连时的鲁棒性, 程序检查前一步是否得到 7 个连通区域, 若不是则重新采用等距划分).

M019301



Figure 19: 7 位码区域

Figure 20: 7 位码分割结果

3.4 数字与字母识别

3.4.1 数据集建立

在得到了序列号的分割结果后, 将每个单独码位裁剪并统一缩放为 32 大小的黑白二值图片存储, 根据提供的注释文件进行标注. 取其中的 85 张火车票作为训练集, 10 张作为验证集, 5 张作为测试集建立数据集 (Fig.21).



Figure 21: 数据集展示

3.4.2 模型结构

模型采用了简单的 CNN 结构, 由 2 个卷积层, 1 个池化层, 3 个激活函数和 2 个全连接层构成, 其中最为重要的特点是模型在预测数字与字母时仅有最后一层全连接层权重不同, 其余各层共享权重, 这在字母数据量不足的情况下对提升模型性能有很大帮助, 因此模型虽然结构简单, 但仍取得了很好的效果, 具体结构如下:

- conv1: in_channels=1, out_channels=10, kernel_size=5 (10, 28, 28)
- pool: kernel_size=2, stride=2 (10, 14, 14)

- conv2: in_channels=10, out_channels=20, kernel_size=3 (20, 12, 12)
- flatten (20*12*12)
- fc: in_features=20*12*12, out_features=512 (512)
- fc_number: in_features=512, out_features=number_class (number_class=10)
- fc_letter: in_features=512, out_features=letter_class (letter_class=26)

3.4.3 预测后处理

由于车票上的 7 位码事实上与 21 位码的后 7 位是完全相同的, 因而在模型对这两部分预测结果不同时需要采取后处理. 考虑到 7 位码的部分可能由于印刷偏差而与左上方的文字重合影响识别, 但由于尺寸较大定位较精准, 且字母处于第 1 位受影响较小, 而 21 位码虽然相对清晰不易受其他印刷字影响, 但字母与数字尺寸相近且均较小, 对字母的分割可能存在不精准的情况, 因此在处理时采用 7 位码预测的字母 +21 位码预测的数字作为最终结果会较为准确 (但在实际测试过程中 100 张图片仅有 3 张预测不一致, 因此这一处理过程也可以忽略).

4 实验结果

4.1 定位与分割结果

本次实验通过两阶段的定位方法实现了对票面与序列号区域的精确定位与分割, 票面的定位过程与票面完整程度无关, 序列号的定位过程基本不受印刷质量的影响, 而序列号的分割也实现了宽度自适应, 对各类字符区域分割得都较为精确. 下图展示了对于各种情况的定位与分割结果: 正常情况 (Fig.22), 印刷模糊情况 (Fig.23), 印刷残缺情况 (Fig.24), 印刷歪斜情况 (Fig.25).

4.2 预测结果

实验采用了 Adam 优化器, 初始学习率设置为 0.001, 在 10 个 epoch 的训练后 (训练时间不足 1 分钟), 数字与字母训练集的 loss 都基本降为 0, 验证集上准确率均达到 100%, 而在测试集图片上也取得了极佳的效果, 仅有 2 位数字预测出错, 且很可能是由于分割不准确导致的, 这证明了我们将数字与字母识别统一于一个模型方法的有效性.

5 分析讨论

尽管在定位与分割过程中都采用了两阶段的精确定位, 但最终结果中仍有极少数的分割不准确现象 (Fig.26, Fig.27), 这可能是由于印刷模糊与二值化阈值不匹配导致的. 实验过程中采用了许多针对数据集的特定参数 (如二值化阈值, 区域边界, 滤波器核与形态学结构元大小等), 虽然在训练集上取得了很好的效果, 但过多的参数可能会降低程序的鲁棒性与普适性, 可以考虑使用泛化性更强的操作完成定位与分割. 同时, 对于训练集中出现较少甚至未出现的字母识别性能无法估量, 可以针对这部分数据修改算法或继续扩充数据集以达到更好的效果.



Figure 22: 正常情况



Figure 24: 印刷残缺情况



Figure 23: 印刷模糊情况



Figure 26: 7 位码分割不精确的情况



Figure 27: 21 位码分割不精确的情况

6 总结

这次课程设计，很好地加深了对课程所讲知识的理解，并懂得了如何在实际项目中加以应用。在项目中，使用了各种图像处理方法，包括二值化，高斯滤波，中值滤波，形态学处理，几何变换等等，同时还尝试了如 Canny 边缘检测，Hough 直线检测等许多其他方法，再加上与深度学习和人工神经网络的结合，是对整个图像处理知识的综合应用，是传统视觉与新兴深度学习结合应用的很好范例，为今后继续在这里领域探索打下了很好地基础。

7 致谢

感谢张超老师和助教们一学期的辛勤付出！