车票序列号检测与识别

——图像处理课程设计报告

熊伟民　1900011621

2022 年 1 月 13 日

目录 (Contents)

1 任务说明

在本次课程作业中，我完成了以下部分的内容：

1、车票票面检测和旋转摆正。

2、21 位码和 7 位码定位与分割。

3、对分割好的数字识别并输出最终结果。

2 运行说明

2.1 运行环境

Windows

2.2 代码结构

```
— training_data/
— dataset/
—— letter/
—— number/
— code/
—— dataset.py
—— predict.py
—— segment.py
—— train.py
—— utils.py
```

2.3 测试方式

将要测试的图片文件夹和包含测试图片名称的 annotation.txt 文件放在 code/文件夹中，在 code/文件夹中打开 Windows 命令行，输入以下命令：python3 predict.py –dir test_data –txt annotation.txt。

输出的车票票面检测、车票序列号分割定位图像保存在父目录的 segments 文件夹中，数字及字母预测结果保存在父目录的 prediction.txt 文件中。

2.4 Github 仓库

代码库已上传至 github: xwm-123/DIP_Project

3 算法原理

3.1 车票票面定位与旋转摆正

在第一个环节，我通过滤波与形态学变换等方法提取出了车票的区域，并使用一些边缘算法提取出了车票所占区域的轮廓，根据轮廓信息将图片中车票区域裁剪出来，并进行摆正。

（1）考虑到车票票面与背景的灰度值不同，我首先对于图片进行二值化处理，采用灰度值 20 作为阈值，得到阈值化后的图片。由于图片中存在噪声，在图片右上角存在白色细线状条纹，因此我采用 5x5 的卷积核对图片滤波得到滤波后的图像，可以清楚地看到，滤波后的图像中已经看不到噪声，只有车票和图片右边缘两块白色的区域。

图 1: 二值化后的图像

图 2: 二值化并中值滤波后的图像

（2）得到滤波后的图像之后，可以看到车票中的文字部分阈值化处理后为暗部，为了去除车票中不连续的文字区域，我对图像进行了形态学处理。我先使用 20x20 的矩形结构元对图像进行闭操作，得到的图像中车票部分均为白色。之后使用 80x80 的矩形结构元对图像进行开操作，去除掉图像右侧的白色区域以及车票下方的两个小矩形区域。

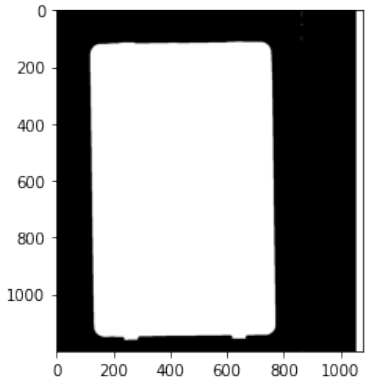

图 3: 开操作后的图像

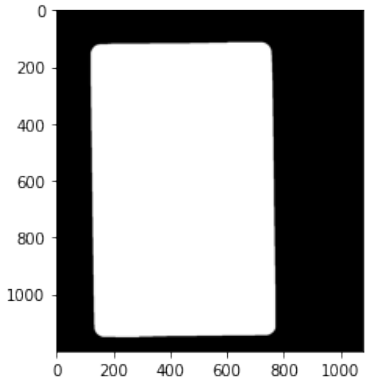

图 4: 开操作和闭操作后的图像

（3）将图像进行形态学处理后，图像中只有车票所在区域为一白色矩形。我使用 opencv 的 findCountours 函数找到图像中的所有矩形轮廓。因为形态学处理后图像中可能还有其他的白色区域，因此选择包含点数最多的结果作为车票的矩形轮廓。之后，我使用 opencv 的 minAreaRect 函数计算车票的最小矩形轮廓，得到包围车票的最小矩形的中心，长宽与相对 x 轴的旋转角度，再使用 opencv 的 boxPoints 函数得到最小矩形的四个顶点，画出了图片中车票的轮廓。

（4）在得到包含车票的矩形长宽，中心与相对 x 轴的旋转角度后，就可以将图像旋转到水平位置，并进行裁剪了。我首先使用 opencv 的 getRotationMartrix2D 函数根据旋转中心和旋转角度得到旋转矩阵，再使用 warpAffine 将图像旋转到水平位置。为了确定旋转后车票的顶点位置，我将旋转矩阵与旋转前矩阵的顶点坐标内积，得到旋转后车票的顶点位置，确定车票的上下

及左右边界，对图像进行裁剪。上面这种方法得到的车票可能是颠倒的，因此我比较车票右下方和左上方区域的平均灰度，当左上方区域的平均灰度更小时，说明车票的二维码在左上方，对车票关于中心做中心对称操作。

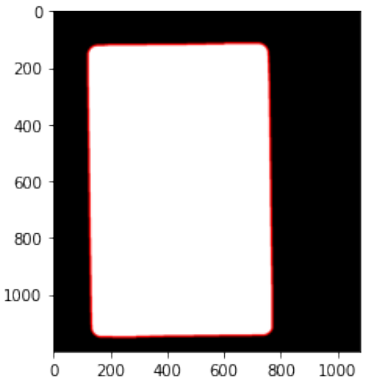

图 5: 车票的矩形轮廓

图 6: 旋转摆正后的车票

3.2 21 位码与 7 位码区域检测与分割

在摆正的车票中，虽然 21 位码与 7 位码的位置可能因为印刷的问题有所差异，但其与二维码的相对位置是基本固定的，因此我先检测容易检测的二维码区域，再根据 21 位码和 7 位码位置与二维码位置的相对关系进行检测。

（1）二维码出现在矫正后车票的右下角，因此我先分割出二维码的区域，对二维码区域使用 20 为阈值进行二值化操作，对于二值化后的图像先使用 45x45 的矩形结构元进行闭操作填补中间的黑色空隙，再使用 60x60 的矩形结构元进行开操作去除右下角的白色区域。将处理好的二维码图像重新放到车票中原来的位置，使用车票边缘检测的相同方法即可得到包围二维码的最小矩形的四个顶点。

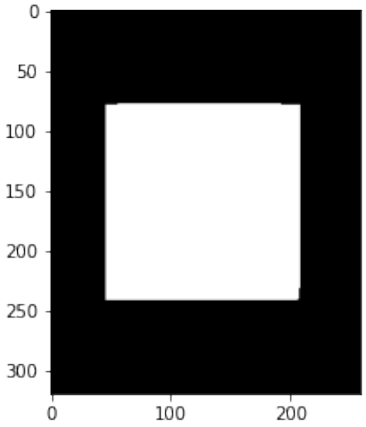

图 7: 形态学操作后的二维码区域

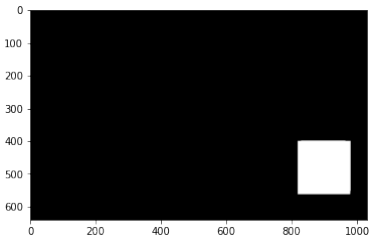

图 8: 二维码在车票中的位置

（2）根据二维码的位置，我粗略计算出了 21 位码和 7 位码在图像中的相对位置，将对应位置裁剪出来后，我采用 20 为阈值将 21 位码的区域阈值化，并提取 7 位码区域中灰度值大于 60 小于 140 的区域作为前景。进行中值滤波去除噪声后，运用与前面相同的形态学方法得到 21 位

码和 7 位码在车票中所在的区域。根据形态学处理的结果，提取出包含这两个区域的最小矩形的轮廓。因为我在在形态学处理的过程中对图像进行了扩张处理以避免不正确的阈值导致的矩形区域的错误，这里我考察了矩形框左右两侧是否有没出现数字的黑色区域，对矩形框进行相应的调整，得到最后的精细检测的矩形框。

图 9: 21 位码的矩形框

图 10: 7 位码的矩形框

（3）对于分割出来的 21 位码和 7 位码区域，我先将对应区域进行二值化和滤波处理，只保留数字部分作为前景。之后使用 findCounters 函数找到所有的连通区域，对应每个数字或者字母，再使用 boundingRect 函数找到包含每个数字或字母的最小正矩形，即可分割出每个字母或者数字，根据划分结果在矩形框中划线。考虑到可能会出现相邻两个数字或字母相连或者单个字符被隔断的情况，我检查连通分量数是否为 21 或者 7，对于不满足的情况采用字母和数字大小比例为 1.5：1 将矩形框分割。

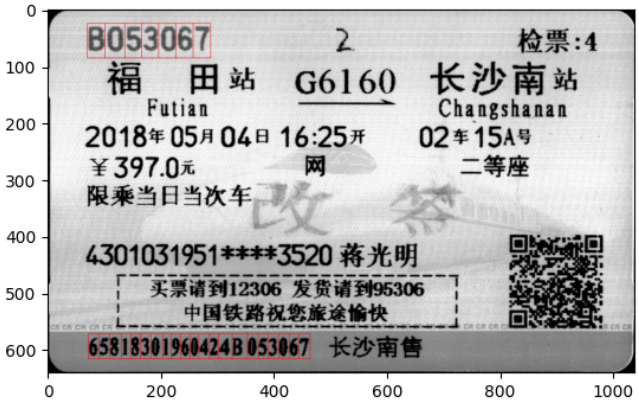

图 11: 划分后的图像

3.3 数字和字母的识别

根据 21 位码和 7 位码的分割结果，我构建了数字和字母的训练集和测试集，分别保存在 dataset/number 和/dataset/letter 两个文件夹中。

我使用卷积神经网络对于数字和字母进行分类，对于数字和字母分别进行训练。对于输入的图片，在预处理部分将图片 padding 为 64x64 作为输入。经过 5x5 和 3x3 的卷积核卷积后，接入全连接网络进行分类。数字图片的训练集图片较多且种类较少，最后在验证集上的准确率

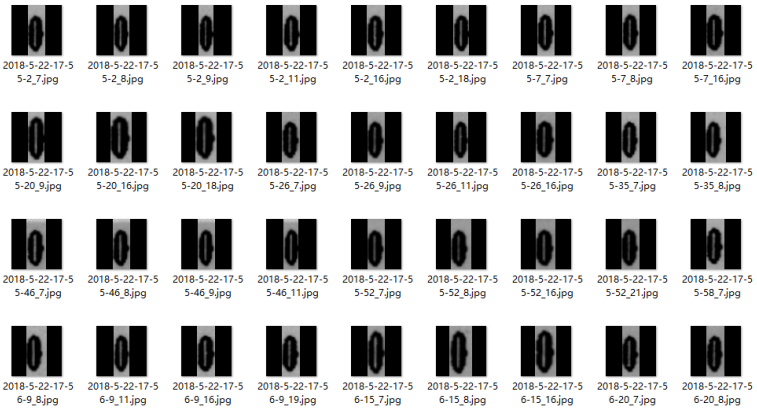

图 12: 部分训练数据

达到了 99% 以上；字母图片的训练集图片较少且种类较多，因此我使用了数字识别网络的卷积层参数，在验证集上也可以达到 90% 以上的准确率。

4 实验结果分析

4.1 结果展示

经过在训练集数据上的测试，我发现算法在大多数图片上都取得了比较好的分割结果，具有很好的鲁棒性，同时对于 21 位码和 7 位码识别的准确率都较高。

图 13: 分割结果

但是也存在着一些问题，例如在图 14 中，左上角的 7 位码分割不准确，存在部分红线穿过图形的现象，经过查看中间的结果得知，因为形态学操作导致图像中出现了不正确的连通分量数，没有进行自适应的分割，而是按照字母与数字宽度 1.5：1 来分割，而 J 字母的宽度相较其他字母较窄，导致出现了不正确的分割结果。需要采用更精细的形态学或阈值处理来解决这一问题。

在图 15 中, 左上角的 7 位码与站名发生重叠，不能正确地根据灰度值分割出 7 位码的区域，导致错误的分割结果。可能需要更多的训练数据和机器学习的方法来消除重叠对于寻找 7 位码

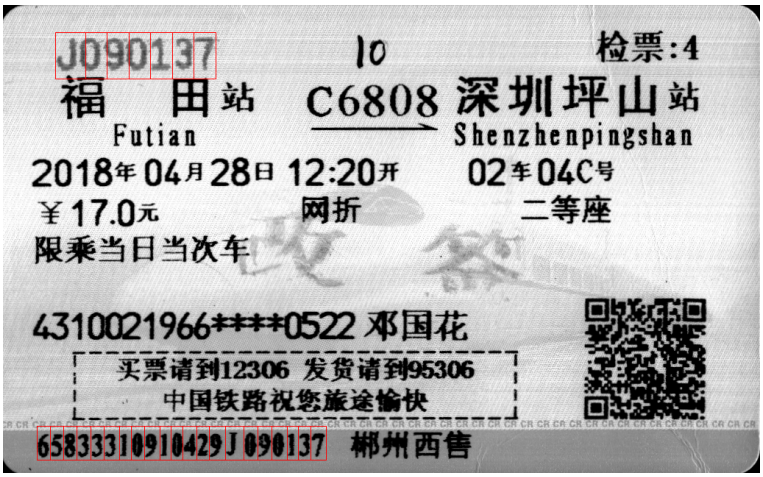

图 14: 不正确的分割结果

区域的影响。

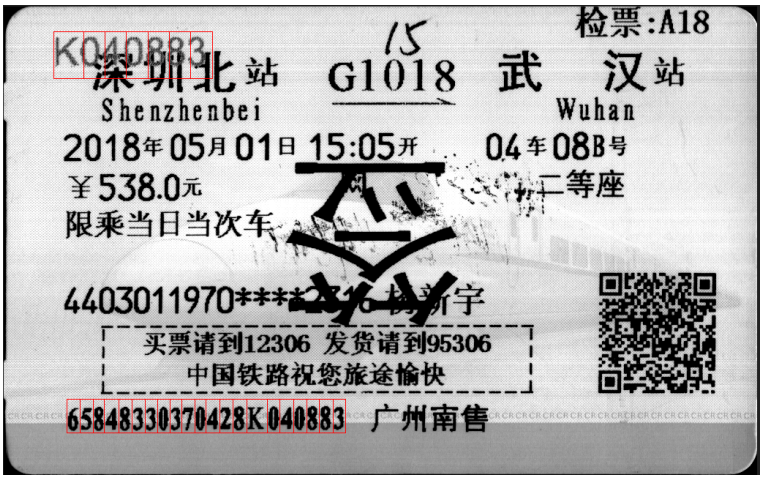

图 15: 不正确的分割结果

4.2 总结与分析

这次车票序列号检测和识别课程大作业帮助我回顾了课程上的知识，同时也让我了解到传统图像处理方法一样可以取得与深度学习相近的结果，且不需要大量的训练数据的支撑。也让我学习到如何根据具体情况选择合适的方法和参数，从而取得好的图像处理结果。