**车票序列号检测与识别**

——图像处理课程设计报告

**1500012741 陈小康**

**1500012797 赵浩然**

**一、任务说明**

1. 检测车票的位置，将它裁剪出后摆正。

2. 在裁剪后的图片中标注出21位码和7位码，并将其分割。

3. 对每张图片，输出对21位码和7位码的识别结果。

**二、运行说明以及代码结构**

* **运行环境**

Ubuntu 16.04、MacOS

* **编程语言**

Python 3.6（外部包请见requirement.txt）

* **代码结构**

—— train\_data/

—— letter\_data/

—— number\_data/

—— code/

———— functions.py

———— mainprocess.py

———— dataset.py

———— evaluate.py

———— mytestdir/

* **测试方式**

请将包含测试图片的文件夹放到 code/ 文件夹中。支持 .bmp, .png, .jpg 格式的图片。然后在命令行中，进入到 code/ 文件夹，输入如下命令：(test\_dir\_name 为测试文件夹名)

|  |
| --- |
| $ python3 evaluate.py --dir test\_dir\_name |

* **Github 地址**

<https://github.com/charlesCXK/DIP-Final>

**三、算法原理**

**1. 车票票面定位**

这部分主要目的是检测车票的轮廓，并进行较为精确的轮廓贴近。

（1）考虑到车票票面与背景颜色不同，首先对整张图像进行二值化。由于四周噪声比较多，二值化后会产生线装条纹，可能影响后期操作。因此，在此基础上使用大小 9\*9 的中值滤波去噪。二值化和滤波后的二值化图像分别见下图。可以明显地看出，去噪后，除了票面区域外几乎无噪点。



图 1 二值化图像

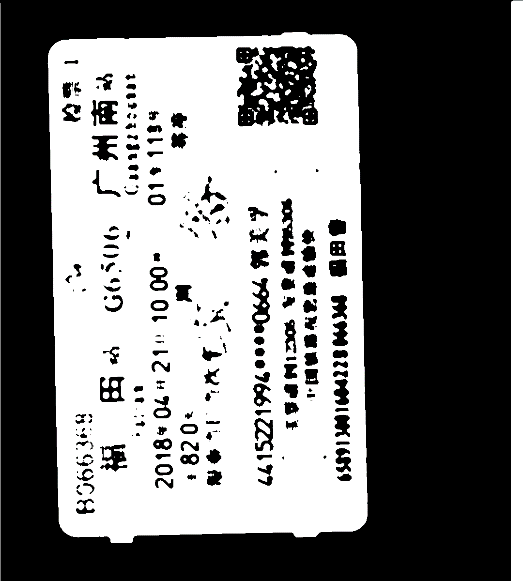


图 2 二值化并去噪的图像

（2）形态学操作

为了能准确检测车票边缘的线，而不是检测出其他乱七八糟的线条，我们需要对车票中间的黑色进行“填充”。最适合的办法莫过于开操作和闭操作了。首先用闭操作填充中间的黑色，然后用开操作“精修”，去除车票四周突出的小矩形（如图2车票下方一排两个小矩形）。处理后的图像如下。

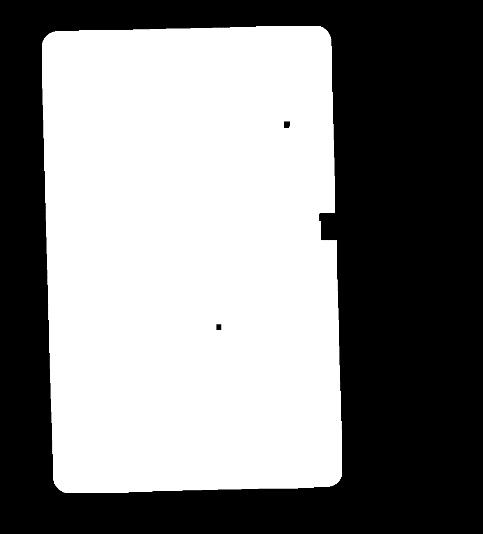


图 3 形态学操作后的车票票面

（3）轮廓检测

先使用 Canny 算子进行边缘检测，然后利用霍夫变换检测直线。之后利用检测到的直线，计算一个最小拟合矩形，以及矩形的四个顶点的位置。由于图中可能存在其他干扰情况，所以最后可能得到多个矩形框，但只保存周长最长的矩形框，作为车票票面。将矩形框画在原图上效果如下：



图 4 车票票面轮廓检测结果

（4）旋转与裁剪

上一步骤已经得到矩形四个顶点，此时可以用顶点计算边的斜率，然后用斜率计算矩形倾斜角，旋转图像使矩形摆正，然后裁剪出来。为了方便后续操作，裁剪后的图像需成为“横长竖短”这样的形状。

对于已经旋转成横长竖短的车票，我们检测其右下角和左上角的灰度值。如果左上角灰度值小于右下角，说明此时二维码在左上角，车票放置反了，我们将其旋转180°回到正常位置。

训练集中有两张图像出现长边缺损情况，我们将这种缺损车票补全到平均长度。我们检测二维码区域（右下角）的平均灰度值，如果偏高说明缺损的是右侧带二维码区域，否则缺失的是左侧区域，我们按照车票平均长宽比，将缺损的区域用白色补全。

**2. 在图片中标注21位码**

（1）二值分割

我们首先将分割好并摆正的车票图像二值化，去除背景和噪音的影响。21位码附近是空白背景，21位码本身是黑色字迹，我们采用固定阈值分割，阈值灰度设为20。二值化结果如下：

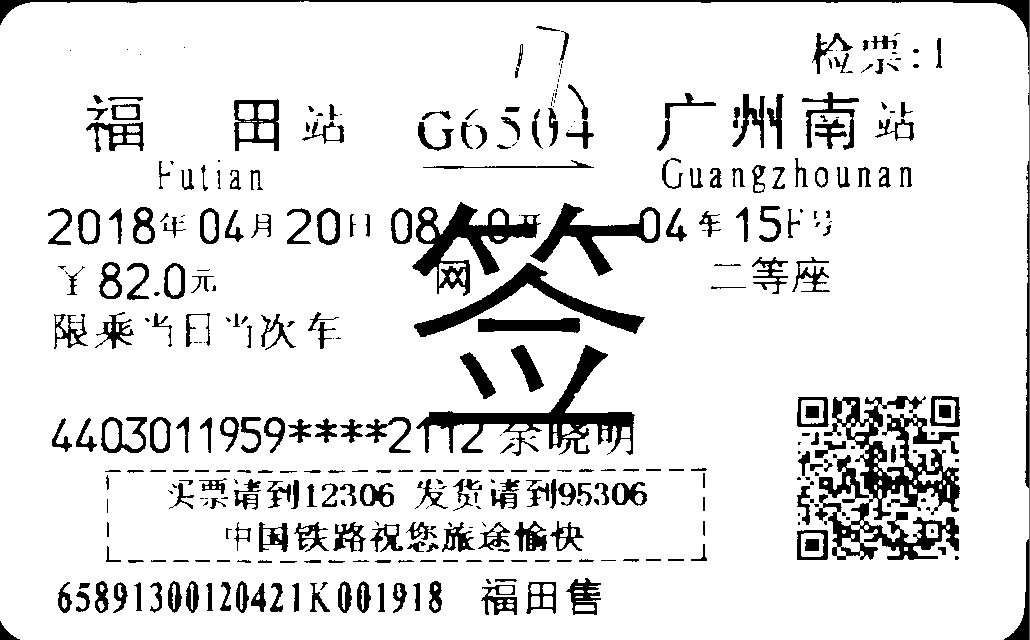


图 5 21位码二值化结果

（2）定位二维码

由于二维码和21位码的相对位置和长宽比比较固定，我们先检测出二维码的高度和位置，以其为标准定位21位码。对未二值化的图像首先进行阈值为20的二值分割，以去除背景和噪音的影响。接下来截取右下角二维码部分，先进行中值滤波去除椒盐噪声，然后取反并做闭操作，使得二维码成为连通区域。

由于训练数据中有一张车票右边缺损的图片，只取左半边的二维码。然后我们根据该区域图片行和，删除底部车票裁剪不精确产生的边缘。接下来将操作后的左半边二维码放回车票对应区域，剩余部分用黑色模板掩膜，随后获得二维码最小矩形闭包的顶点，通过顶点确定二维码高度像素值。

（3）获取21位码区域

经过对图片的测量，我们确定了二维码高度和像素的映射关系，二维码高度和二维码左边缘到21位码左侧、顶部的长度比，以及与21位码长宽的比例。由此根据上一部分获得的二维码高度像素值，可以确定21位码的大致范围。

我们对这一区域先取反，然后做闭操作和膨胀操作，让21位码形成连通区域，同时避免框得太紧。然后将处理后的结果放回车票对应区域，其余区域用黑色背景掩膜。接下来获得最大连通区域的最小矩形闭包的四个顶点，将其作为一个返回值。另一个返回值是画好了21位码区域的车票，这个值主要是在编程过程中查看中间结果用。

**3. 在图片中标注7位码**

获得7位码和获得21位码的操作过程相似。

（1）获得7位码大致区域

和基本为纯黑色的21位码不同，7位码灰度值较大，我们采用双阈值分割，将60 ~ 120灰度值之间的像素值设为255，其余区域设为0，相当于在二值化的同时做了取反处理。

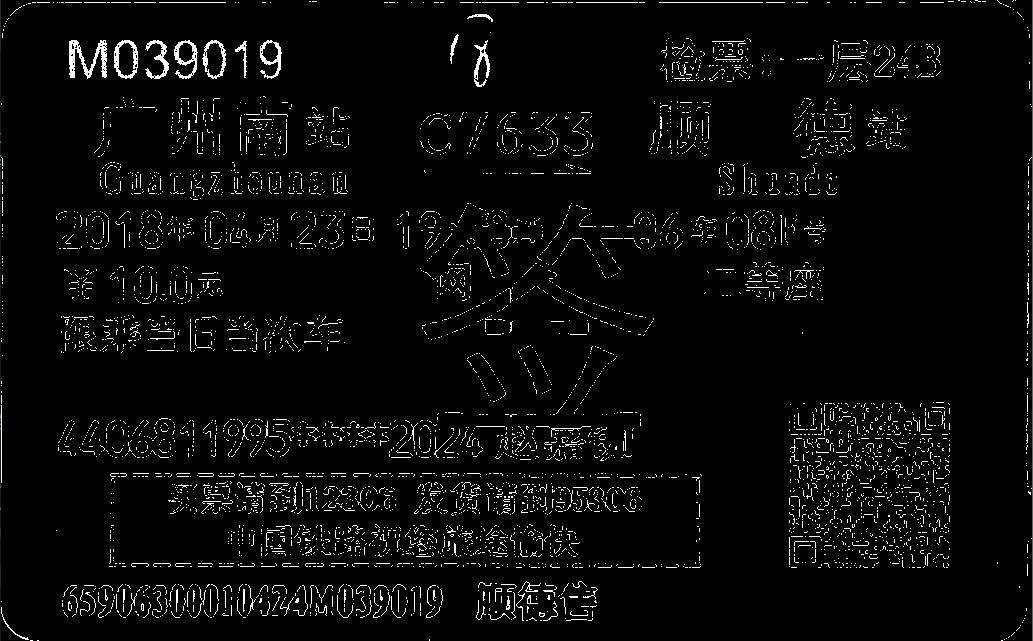


图 6 7位码二值化结果

接下来我们参考21位码部分的操作，获取二维码闭包顶点，根据二维码高度和与7位码距离、7位码长宽等信息裁出7位码大致区域。

（2）精确定位7位码

与21位码不同，7位码的印制区域误差比较大，有些还和下面的车站信息重叠到一起。在二值化并裁减之后，保留的是7位码和车站名汉字的部分轮廓。

我们首先尝试用均值滤波和阈值分割去除汉字，这一步骤后汉字轮廓大部分被去除，对残留的类似盐噪声的白点，再用中值滤波将其去除。然后同样用闭操作好膨胀操作使得7位码区域连通，同时防止框得太紧。接下来参考21位码检测步骤，对处理后的区域掩膜并放回原图，获得最大连通区域的最小矩形闭包的四个顶点，和画出了7位码区域的图片一起作为返回值。



图 7 均值滤波结果

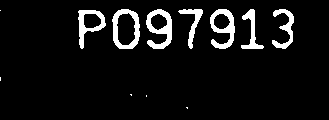


图 8 中值滤波结果

**4. 票面标注与分割**

这一部分根据以上两个类返回的矩形闭包顶点，在车票图片中对应位置框出两组码并分割。输入也是分割好并摆正的车票，另外还有21位码和7位码的最小矩形闭包顶点。

首先将输入的灰度图像转化为RGB彩色图像。然后调用两次drawSegLines函数先后画出21位码和7位码的区域，并分割各位数字/字母。最后将用红线标出分割结果的图片返回。

drawSegLines函数输入的是裁剪好的车票图片，以及闭包顶点和数字+字母总数。

（1）重新排序顶点集合

首先判断输入的是灰度图还是RGB彩色图，如果是灰度图转换为彩色图。然后将四个顶点按照左上角、左下角、右下角、右上角的顺序排列。此处由于车票已经摆正，矩形长边的斜率一般都很小，所以将横纵坐标之和最小的点作为左上角，然后根据与左下角横纵坐标之差依次定位其它点，按顺序生成新的顶点集合。然后计算长边斜率。

（2）删除左右白边并画出外轮廓

按照上面的顶点集合取出图像对应区域，取反后删掉左右两边灰度值和为0的背景列。将顶点集合的坐标根据缩减的列数相应变化，然后画出外轮廓。

（3）画出内部分割码字的线

接下来我们根据分割数在内部同样用红线分割码字。由于字母和数字的宽度不同，我们要判断是7位码还是21位码，对其中字母部分额外增加宽度。经过测量和调参我们确定字母宽度约为数字的1.5倍，我们令长边列数/21.5或长边列数/7.5为一个长度单元，分别以左下角和左上角为起点，在横轴上每次增加一个长度单元，到字母位置后增加1.5个长度单元，根据斜率取各自长边上对应位置的点并连线。这样就把21位码和7位码根据位数不同分割了。



图 9 21位码分割结果



图 10 7位码分割结果

**5. 数字与字母的识别**

利用21位码和7位码检测结果，按照数字和字母的比例，将其切割成小图片。由于只能按整数像素个数为单位长度来切割，所以要先将图片调整大小，保证能够恰好将每一个数字和字母切割出来。切割出来之后的单个数字/字母以及它们对应的标注分别存放在 number\_data 以及 letter\_data 文件夹中，如图所示：

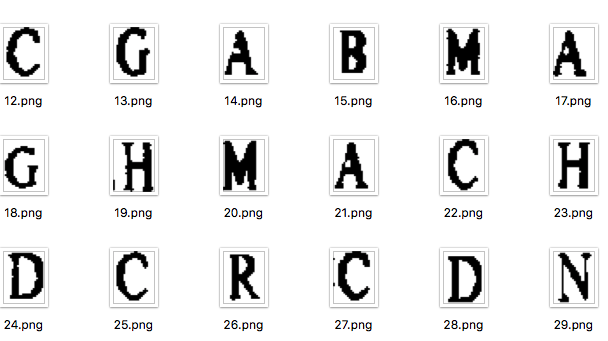


图 11 字母数据



图 12 数字数据

对于每张图片，将其展开成一维向量作为图片的特征值。考虑到数据量很小，我们利用线性 SVM 为数字和字母分别训练分类器。在训练时，采用 70% 的数据作为训练集，30% 的数据作为验证集。由于数字样本很多，所以最后在验证集上的准确率可以达到 97%~99%；而字母数据样本很少（总样本只有100），却有26个类别，因此准确率相应地会降低很多。在多次测试时，准确率大概在 70%~90%之间波动。

**四、实验结果及其分析**

经过对全部图像的测试，我们发现我们的算法对图像的各种异常情况有一定的鲁棒性，比如车票票面不完整、车票票面信息印刷倾斜等问题，都能较好地处理，但还是存在一些不足。下面，我们将以几张图为例，详细说明目前算法的表现。



图 13 分割很好的情况

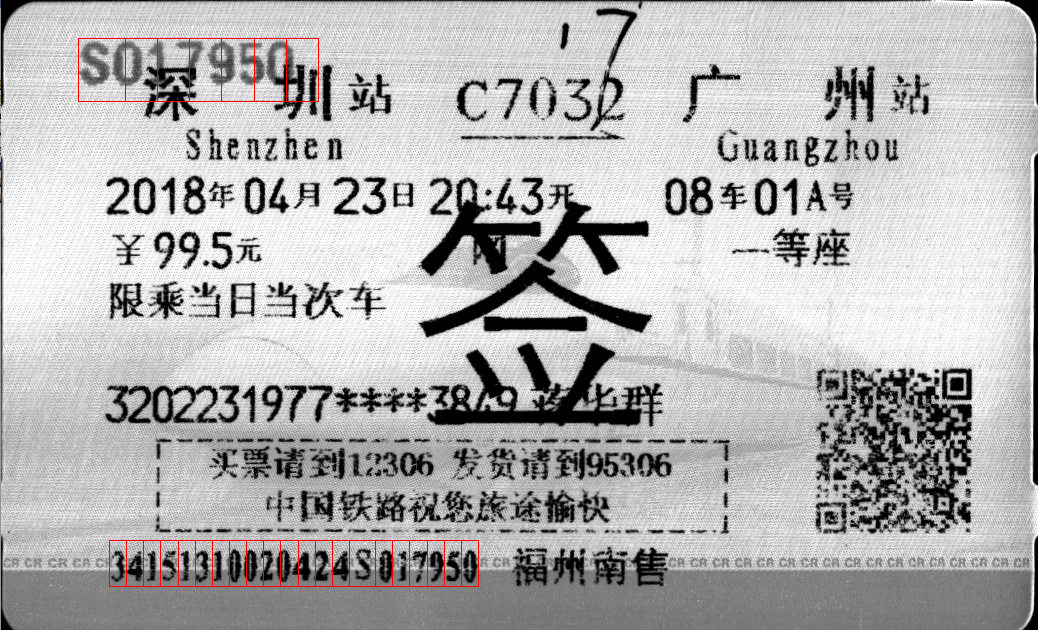


图 14 印刷倾斜的情况

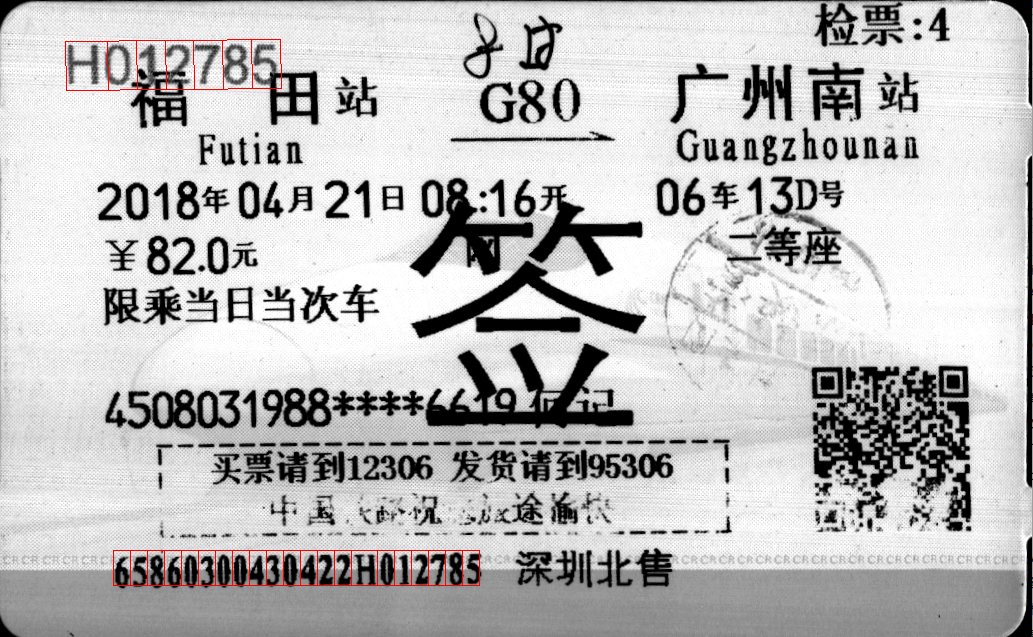


图 15 二维码缺失的情况



图 16 分割不精确的情况

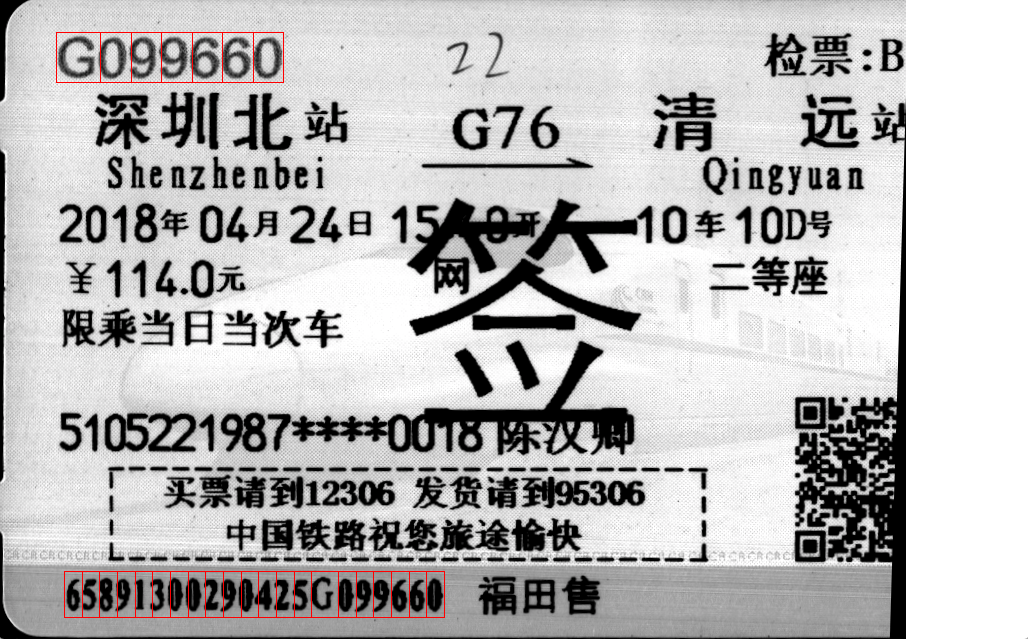


图 17 车票缺失的情况

首先请看第一张图（图13），这是算法表现很好的一个示例。由于7位码和21位码分布的区域很规则，因此都较好的分割了。

请看第二张图（图14）。这里的21位码区域整体偏上，超出了车票底部的深灰色区域。由于我们采用二维码的位置对21位码区域进行定位，所以我们仍然检测出了21位码的区域，并成功分割。但是，由于这张图的7位码区域和汉字有重合，所以7位码的分割比较失败。

请看第三张图（图15）。这张图上，二维码部分缺失，处理后下面可能少一部分，因此理论上会对我们定位21位码的区域造成一定影响。但我们的算法仍然较准确地框出了21位码的位置。

请看第四张图（图16）。这张图的21位码有几个元素被框的比较紧，思考后总结出主要原因是：1、26个字母不一定等宽，按照同一个比例来划分字母和数字，可能会造成某些数据的不适应。2、数字占据的空间不一定等宽，比如数字“1”占据的空间比其他数字要小。

请看最后一张图（图17）。这张图车票票面只有一部分，我们算法自动对其补全，最后的分割结果也比较理想。

**五、实验分工**

陈小康：

整体框架结构设计；车票票面定位与裁剪；数字、字母的识别；评测接口编写；对21位码和7位码的检测提供一些微小思路。

赵浩然：

负责21位码和7位码的标注与分割；将横长竖短的车票摆正并补全；对裁切出来的码字区域删除多余边缘。

**六、总结**

这次大作业，加强了对课上所学图像处理知识的理解，懂得了在某种特定情境下，该运用什么样的图像处理手段来得到预期的图像。这次大作业用到的知识点，横跨课上讲的很多方面，是一次综合性的实践。