北京大学

《图像处理》大作业报告

车票序列号检测与识别

姓名：董欣然

学号：1900013018

课程：图像处理

任课教师：张超

一、任务说明

在本次课程作业中，我完成了以下任务：

1) 车票票面裁剪、旋转摆正。

2) 21 位码和7 位码区域精准定位。

3) 21 位码和7 位码分割。

4) 训练模型，字母和数字识别。

二、运行说明

1) 环境说明

运行环境

Windows10

语言版本

Python 3.8.5

模块版本

torch==1.8.0+cpu

opencv-python==4.5.3.56

numpy= 1.19.2

matplotlib=3.3.2

tqdm= 4.27.0

2) 作业说明

提交的作业包中含有课程设计报告和root文件夹，root文件夹包含了作者的项目源文件，文件夹结构如下：

- root/ 根目录
  - image_process.py 用于票面分割、识别的代码
  - model.py 神经网络模型
  - train.py 训练模型的代码
  - model.pth 神经网络模型
  - prediction.txt 测试输出文件
  - segments/ 测试输出文件夹，用于存放票面分割图片
  - test_data/ 测试输入文件夹，用于存放测试图片
    - annotation.txt 测试输入文件

3) 运行方法

将测试图片和 annotation.txt 放入 test_data/文件夹中，在根目录 root 打开命令行，键入“python image_process.py”。输出文件 prediction.txt 将保存在根目录 root 下，票面分割的图片将保存在 segments/文件夹中。

三、算法原理

1) 车票票面检测

作者通过阈值处理与形态学算法提取出了车票所占区域的轮廓，根据轮廓信息将图片旋转、裁剪，获得只含车票的矩形图像。具体操作如下：

1. 阈值处理

采用全局最佳阈值处理方法将图片二值化（如图所示①）。

2. 形态学处理

首先对图片进行开操作，消除票面以外的噪声。再对图片进行闭操作，消除票面上的文字，最后对图片进行开操作，消除票面边缘的凸起（如图所示②）。

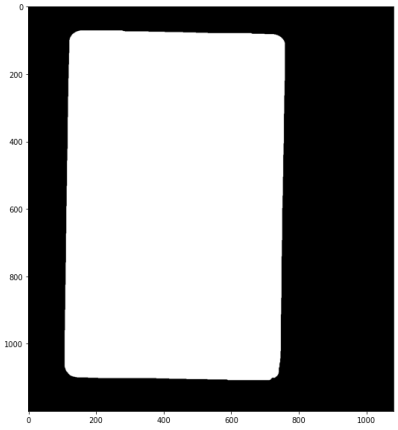
3. 旋转裁剪

通过边缘检测技术找到二值图片上最大的矩形区域，将矩形旋转摆正，并进行裁剪。

4. 翻转

由先验知识可知：票面下部分的平均像素值低于票面上部分。根据票面的灰度值判断票面方向，并进行翻转。效果如图③所示。

图①



图②

图③

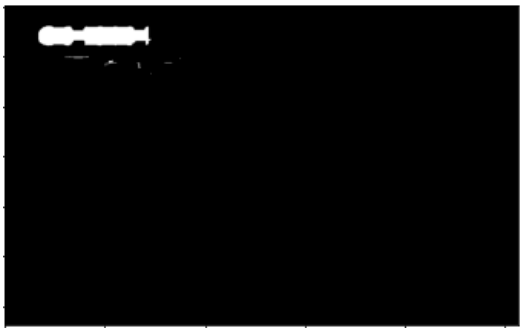
2) 7位序列码分割

1. 首先对票面进行二值处理。可以发现：当阈值在限定范围内变化时，除了七位序列码之外的区域基本没有变化（如图④⑤所示）。故作者采用二值化图片差分的方式提取出七位序列码基本形状（如图⑥所示）。
2. 对于差分后的图片，采用形态学方法把序列码所在区域进行腐蚀和闭操作，此时二值图片的序列码位置变为一个接近矩形的白色区域（如图⑦所示）。计算包含白色区域的最小矩形，矩形区域可认为是7位码的精确定位。

图④

图⑤

图⑥



图⑦

3. 使用findCounters 函数找到矩形区域中所有的连通块，理想情况下每个连通块对应一个数字或者字母。
4. 使用boundingRect 函数找到包含每个连通块的最小正矩形，即可分割出单个字母或者数字。

5. 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，检查连通分量数，对于连通域数量不等于7的情况，作者采用字母和数字大小比例为1.5：1 将矩形区域分割成七块。

7位序列码分割效果如图⑧所示。

图⑧

3) 21位序列码分割

21 位序列码分割方法和 7 位序列码类似。21 位序列码区域的灰度值较小，故考虑使用较小的阈值（作者设置阈值为 30）进行二值化处理，通过形态学方法可定位出 21 位序列码精确区域。通过查找精确定位区域的连通块个数，可以确定 21 位码的分割方案。对于连通域数量不等于 21 的情况，作者采用字母和数字大小比例为 1.5：1 将矩形区域分割成 21 块。

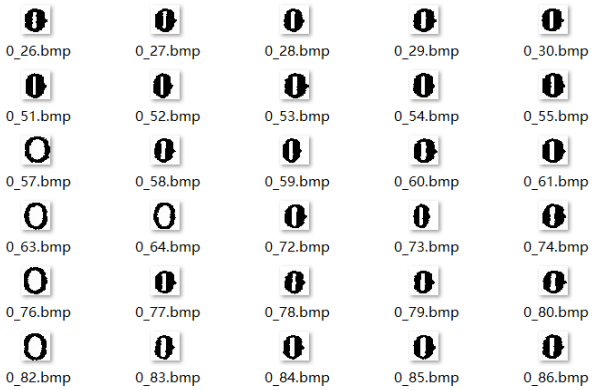
21 位序列码分割效果如图⑨所示。

图⑨

4) 数字与字母识别

1．数据集建立

得到序列号的分割结果后，将每个字符裁剪并统一缩放为32大小的黑白二值图片存储，根据提供的文件annotation.txt进行标注．取其中的90%火车票作为训练集, 10 %作为验证集。数据集展示如图⑩所示：



图⑩

2.模型结构

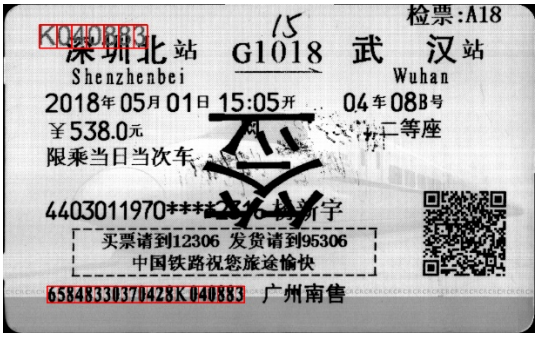
模型采用了简单的CNN 结构，由2 个卷积层, 1 个池化层, 3 个激活函数和2个全连接层构成，其中最为重要的特点是模型在预测数字与字母时仅有最后一层全连接层权重不同，其余各层共享权重，这在字母数据量不足的情况下对提升模型性能有很大帮助，因此模型虽然结构简单,但仍取得了很好的效果。神经网络结构如下：

```
self.conv1 = nn.Conv2d(1, 10, kernel_size=5)    # (1, 32, 32) -> (10, 28, 28)
self.pool = nn.MaxPool2d(2, 2)                  # (10, 28, 28) -> (10, 14, 14)
self.conv2 = nn.Conv2d(10, 20, kernel_size=3)   # (10, 14, 14) -> (20, 12, 12)
self.flatten = nn.Flatten()                     # (20, 12, 12) -> (20*12*12)
self.fc = nn.Linear(20*12*12, 512)              # (20*12*12) -> (512)
self.fc_number = nn.Linear(512, number_class)   # (512) -> (number_class=10)
self.fc_letter = nn.Linear(512, letter_class)   # (512) -> (letter_class=26)
```

四、实验结果展示

本次作业设计的算法在大多数图片上都取得了比较好的分割结果。对于普通票面的21位码和7位码分割效果很好（如图⑪所示）。即使是在七位码印刷位置和站点名称印刷位置严重重合的情况下，也可以完美地分割图像（如图⑫所示）。

图⑪



图⑫

但是也存在着一些问题，例如在图⑬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。

图⑬

五、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，同时也让我认识到：巧妙地使用图像处理方法可以大大提升深度学习的效率，也丰富了我深度学习的实操经验，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。

参考文献

[1]《数字图像处理（第三版）》，（美）冈萨雷斯，（美）伍兹
[2] https://github.com/eastmountyxz/ImageProcessing-Python