车牌识别系统说明文档

一、综述

本次车牌识别系统针对中文车牌开发设计，并将整个识别过程划分为三个子模块，即车牌定位，字符分割以及识别（包括颜色识别和字符识别）。在三个子过程中采取了多种优化方法与处理策略，尤其是车牌定位和字符分割部分，关系到开发过程中实际数据与训练数据的生成，因此会直接影响到模型的准确性，以及模型判断的最终结果。目前，在比赛给出的数据集上，对大部分模块都达到了较高的识别率，在省市简称子库、平均亮度变化子库，亮度不均匀变化，错切变化，透视变化等子库上均实现了超过90%的识别率，总体识别率在80%左右，在车牌种类变化子库中表现欠佳。

二、开发环境

Windows10

Opencv3.4.1

Python3.6

Tensorflow1.8.1

三、工程结构

在本系统中，代码结构如下图：

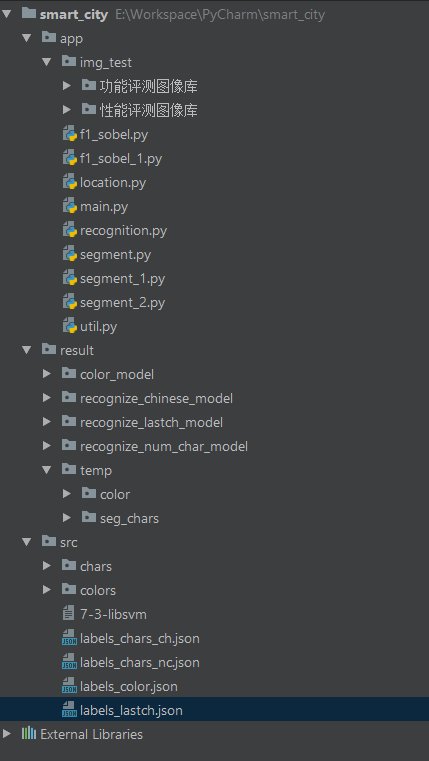


图1-1 代码结构图

目录app下存放系统的所有代码，目录img\_test中存放本次比赛提供的测试图片数据;目录result中存放颜色识别及字符识别的相关训练模型，目录temp下则存放系统识别车牌过程中生成的中间数据如切割好的字符;目录src存放训练过程中的json标签文件，训练字符和颜色的原始数据集，以及定位过程中SVM判断车牌训练参数。

四、系统运行流程

4.1.车牌定位

车牌定位模块的主要实现思想为，首先通过一定的策略与方法检测所有可能为车牌的图片轮廓，然后通过训练svm得到的svm模型来对每个轮廓是否为车牌进行判断，最后输出是车牌概率最高的轮廓图片。

轮廓筛选主要采取了三种定位策略：颜色定位策略、边缘检测策略以及MSER检测策略。即通过车牌的颜色特征，字符边缘特征以及阈值变化特征三种特征来检测所有可能为车牌的轮廓。然后对获取到的轮廓进行初步的筛选如面积筛选，长宽比例筛选，旋转角度筛选，根据给定的车牌数据可知旋转角度小于30度（旋转变化子库除外），面积与长宽比也可根据先验知识设定阈值。第一轮筛选过后，再利用svm进行最终判断，大大减少计算量的同时也能保持极高的准确率。

具体流程如下：

1. 读入图片
2. 灰度化
3. 高斯模糊
4. 二值化
5. 颜色检测
6. Soble算子
7. MSER检测
8. 轮廓筛选
9. svm判断
10. 输出结果

4.2 字符分割

对于车牌定位到的车牌图片，在作字符分割之前要进行一定的处理，以便后面的字符分割能够顺利进行。

首先处理车牌的上下边框及铆钉，此部分采用的方法为：对二值化后的车牌像素进行逐行的分析，计算其像素跳变次数，字符部分的跳变次数较高，而边框部分的跳变次数较低，对跳变次数低于阈值的行，将其像素值设定为0，即达到了去除上下边框的目的。

其次要处理车牌中的小块噪声，比如在车牌第二个字符与第三个字符中的圆点，因为经过了上下边框的处理，字符的横向粘连情况已经降到最少，可以直接统计x方向上每一部分连续像素的面积大小，面积小于标准字符的，将其像素值设定为0，即可以达到去除小块噪声的目的，此时得到的二值化车牌图片基本上只有独立的字符。

字符分割模块采取的主要策略是投影切割，即对定位好的车牌图片进行一定处理后二值化，然后对二值化的图像进行垂直投影统计x坐标上每一列的像素个数，根据车牌上字符的形状特征先验知识可知，字符间的间隙对应的x坐标上像素个数应该是极少（理想状况下为0），因此根据对垂直投影像素个数的判断可以获取字符的开始位置及结束位置，然后进行字符切割即可。

具体流程如下：

1. 取得定位后的图片
2. 处理上下边框
3. 去除小块噪声
4. 分析垂直投影像素数据获取字符位置
5. 字符切割
6. 输出字符个数正确的结果

4.3 识别

识别模块采用深度学习CNN卷积神经网络LeNet-5架构，分别对车牌颜色、车牌汉字、车牌数字及字母进行识别。其模块主要分为数据提取与图像识别两部分，具体流程如下图所示：



图4‑1 识别流程

1.数据提取

数据提取是用于为识别模型提供数据集，针对不同的训练模型，读入不同的数据集，对外接口如下图所示：



图4‑2 数据提取

2.图像识别

车牌图片经前期处理后，送入识别模块。在模型存在的条件下，直接调用对应的模型，对图片进行识别，并对不同模型产生的结果进行整合，得到最终结果。流程如下所示：

图4‑3 识别流程