第三次竞赛说明文档

一、用到的工具

Python的图像处理模块pillow和OCR模块pytesseract

二、大致思路

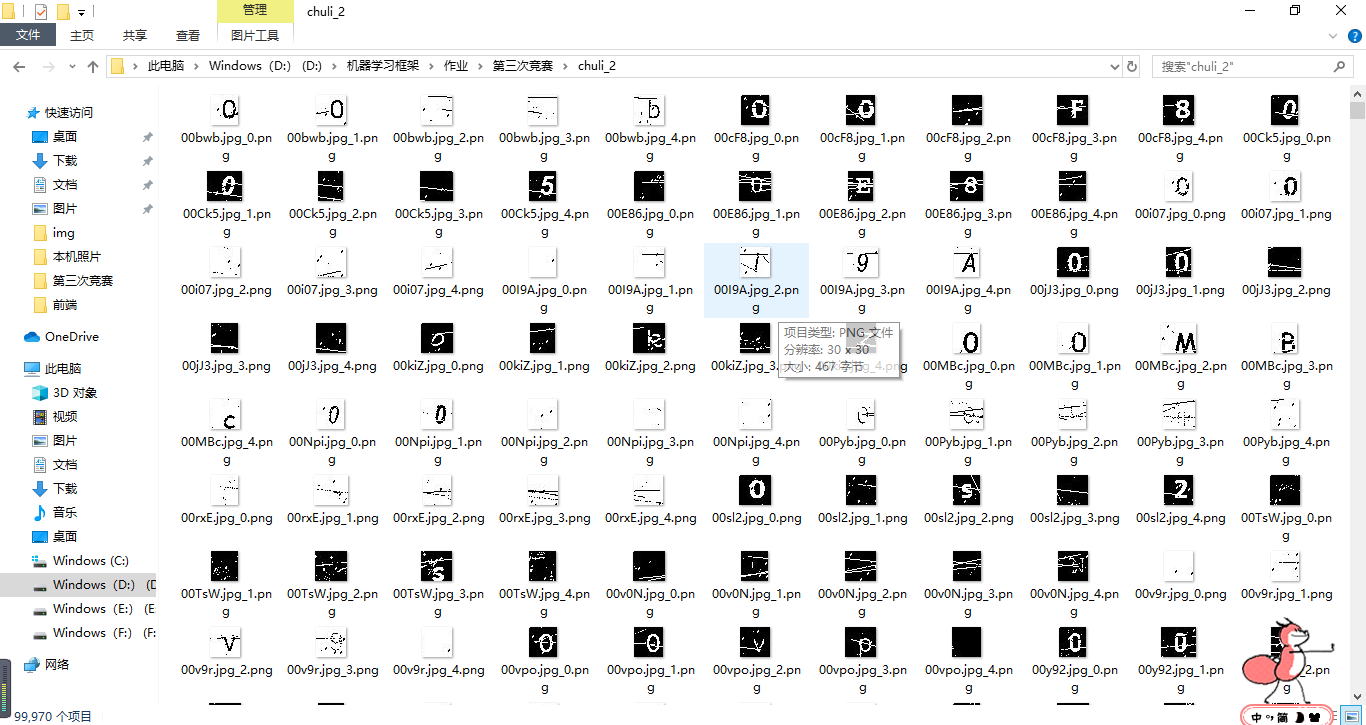
一共有20000张训练集和20000张测试集，训练集与测试集都是彩色图片，并且存在噪声和干扰线条，我认为首先应去除干扰项，比如进行去噪、灰度化、二值化等操作。测试集和训练集集都进行上述操作之后，对训练集进行近一步的操作。通过观察，每张图片都含有五个字符，这些字符有数字和字母（存在大小写的区别），可以将每张图片都进行切割，再通过识别每张小图片上的字符进行分类，可以用KNN算法进行分类。最后根据已有的学习到每个字符的存在形态对测试集进行识别，得到测试集的验证码。

三、瓶颈期的苦苦挣扎

对训练集进行了一系列的灰度化、去噪、二值化的处理，效果如下：



结果不是很理想一些图片完全看不清楚，之后我依旧对图片进行了切割，效果如下：



分割之后效果还可以，每张图片上都有字符。再下一步打算进行分类处理。可想而知分类结果十分差强人意，很多图片根本不能正确分类，为后续的学习带来了很大的隐患。我用的KNN算法进行分类，之后又用了决策树，结果也很不好。之后我在这个地方卡了较长的一段时间，最后忍痛割爱，打算换一种方法。毕竟作业有截止时间，我打算后续再用别的分类算法继续进行分类学习，现在面临的问题是我要在有限的时间内找到最优的方法。

四、新的思路

图片的处理是肯定需要的，接着我想利用pytesseract模块进行验证码的识别，去掉识别结果中的特殊字符，获得识别结果。在下载安装Tesseract过程中也遇到了一些问题，路径问题，不兼容问题等，不过最后成功解决了。很显然，新的方法也不是很成功，对测试集进行去噪、二值化处理后的效果如下：



最后因为时间问题，依然选择的这种方法。

五、算法过程

1.将训练集图像进行灰度处理，转化为灰度图像；

2.获取图片中像素点数量最多的像素（此为图片背景），将该像素作为阈值进行二值化处理，将灰度图像转化为黑白图像（用来提高识别的准确率）；

3.去掉黑白图像中的噪声，噪声定义为：以该点为中心的九宫格的黑点的数量小于等于4;

4.对训练集进行训练学习。

5.重复上面的前三步对测试集进行处理。

6.利用pytesseract模块识别，去掉识别结果中的特殊字符，获得识别结果。

六、代码中一些函数的简述

get\_threshold()函数获取图片中像素最多的像素，通过字典的形式记录像素出现的次数，最后把出现次数最多的像素点设为背景。

get\_bin\_table()函数是对图片进行二值化处理，通过像素阈值在适当范围内进行处理。

Cut\_noise()函数用来去掉二值化后的图片中的噪声点，对图片的边缘同时也需要进行一些处理，把噪声处的像素置为背景像素。

OCR\_lmj()函数是调用tesseract.exe对处理后的图片进行识别，并返回识别结果。

七、心得与体会

这次竞赛过程中遇到的困难比前两次多得多，代码的思路比较混乱，我整个人的思路也混乱，得到的结果也不是很满意，但是在解决困难的过程中学到了不少东西，一遍一遍的尝试，一遍一遍的更正，自己真的收获了很多，在后续的时间里还要对这次的竞赛代码进行改进，寻找更好的解决办法，希望能够得到更好的结果，收获更多的东西。