基于简单 KNN 的北大教务部网站验证码识别

——数据结构与算法实习大作业

何昊 1600012742 hehao1998@pku.edu.cn

徐雪桥 16000xxxxx 16000xxxxx@pku.edu.cn

一、概述

这次验证码识别大作业，我们选择北京大学教务部网站登录页面的验证码作为识别的对象，完成了图像预处理、构建数据集、构建测试集、数据集和测试集标记、以及基于 KNN 分类算法的验证码识别，通过一些优化，达到了 98.25%的字符识别准确率和 93%的整个验证码图片的识别准确率。

二、系统设计概述

语言	Python 2.7	
支持的系统	Mac OS X 10.12，python 解释器版本 2.7.10 Windows 10，python 解释器版本	
第三方库	科学计算库 Numpy 图像处理库 Pillow GUI 库 Tkinter 网络库 urllib2	
可执行源代码文件	get_data_usage.py	从教务官网上抓取数据集的脚本程序
	proc_image_usage.py	用来对抓取数据进行图像预处理
	gen_label.py	简化人工标注数据集过程的脚本程序
	knn.py	命令行程序： 功能是读取 train 文件夹下训练集使用 KNN 分类算法训练，然后读取 test 文件夹下的测试集测试，输出测试结果。
	knn-gui.py	GUI 程序： 可以读取 train 文件夹下训练集使用 KNN 分类算法训练，然后动态地即时从教务网站获得并显示图片，输出识别结果。
辅助库	get_data.py	一个爬虫抓取数据的函数
	proc_image.py	一些用于预处理源代码图片的函数
	dataset.py	以矩阵描述数据集的对象
文件夹	data/	1000 张由 get_data_usage.py 抓取的验证码
	train/	共 400 个经过标记的单个字母图片， proc_image_usage 预处理完后进行了人工标记
	test/	100 张由 get_data_usage.py 抓取的验证码，人

		工标记了正确答案
	train-1label/	第一次尝试所使用的训练集，其中没有对粗体和斜体进行分类
	document/	与文档有关的一些文件
	debug/	临时文件夹，可忽略

三、数据的获取

1、解决思路

北京大学教务部的网站登录界面网址是：http://dean.pku.edu.cn/student/，其中在登录阶段需要输入形如  这样的验证码。

通过查看网页源代码，发现验证码的生成网址为：

http://dean.pku.edu.cn/student/yanzheng.php?act=init

且每次刷新，生成的验证码都不同。

因此，使用 Python 创建简单爬虫程序即可完成数据集获取。

2、程序设计思路

定义函数 getdata(num, folder)，其中参数 num 为抓取的图片数量，folder 为存储的文件夹名称。如果文件夹在执行文件根目录下不存在则创建文件夹，之后从网址抓取图片存储在文件夹中。图片依次命名为 1.png、2.png，……，num.png。

核心爬虫代码：

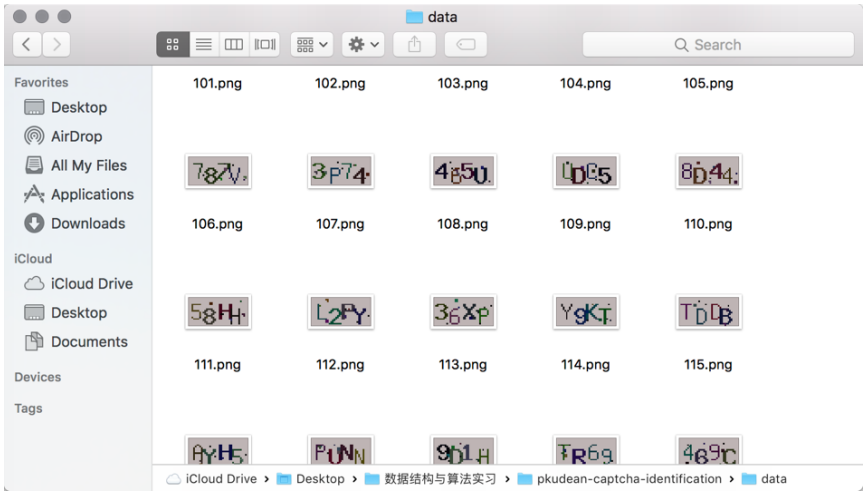
```
im_url = 'http://dean.pku.edu.cn/student/yanzheng.php?act=init'
im_data = urllib2.urlopen(im_url).read()
f = open(folder + os.sep + str(cnt) + '.png', 'wb')
f.write(im_data)
f.close()
```

3、代码文件

get_data.py: getdata 函数的实现

get_data_usage.py: 可执行文件，使用 getdata 函数完成数据集的抓取

4、运行结果：data 文件夹下的 1000 张图片和 test 文件夹下的 100 张图片



四、训练集的构建

1、实现目标：

将得到的验证码图片中的每一个包含四个字母/数字的图片分离成 4 个图片，每个图片包含一个字母/数字，颜色上进行二值化只保留黑白两色，便于进行训练。

2、解决思路

观察诸如  这样的验证码，发现首先图中有一些干扰色块，但是干扰色块的颜色一般与字符不相同，那么如果一个色块为图片中的一个连通分量的话，利用广度优先搜索，寻找大小小于某个阈值的连通分量的色块，然后就能去除这些干扰色块。之后分析其中的字符，发现每个字符的颜色各不相同，但是每个字符不一定是连通的分量。因此考虑统计一张图片中不同颜色的出现频率，频率最高的为背景色，排在第二到第五的就是四个字符的颜色了，然后根据不同颜色出现的位置来判断这四个字符的顺序。对所得的每个字符，将其分离成单独的图片，剪裁成统一的大小，输出到文件夹中。

3、程序设计

主函数：

proc_image(src_folder, dest_folder, image_num)：从 src_folder 中读取 1.png, 2.png, …, image_num.png 图片，将这些图片分离后的结果按从 1.png 开始顺序编号存储到 dest_folder 文件夹中。

split_image(im)：输入一张验证码图片 im，返回分离后的四个字符图片的列表。用于最终测试。

辅助函数：

remove_noise(image)：输入一张图片 image，返回其中的小于一定阈值的色块被去除设为背景色的图片。

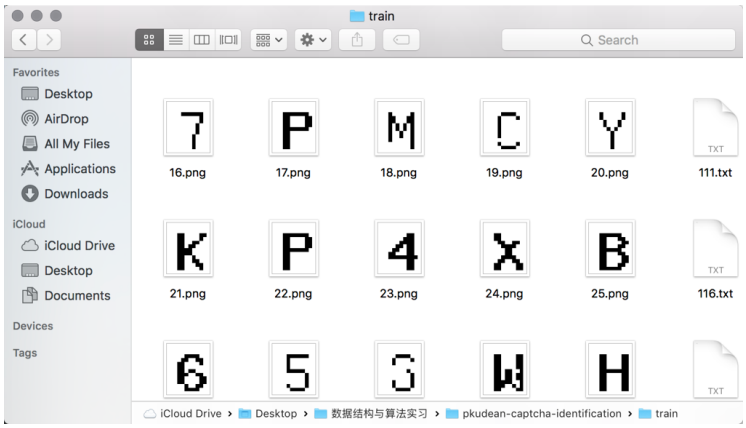
compute_occurences(image)：输入一张图片 image，返回描述其中的颜色的出现频率的二元组[color, cnt]的列表。

4、代码文件

proc_image.py:　　存放与处理图片和构建训练集有关的函数

proc_image_usage: 使用 proc_image.py 内的函数完成对 data 文件夹下图像的处理，输出 400 张经过处理的单个字符图片到 train 文件夹中。

5、运行结果



五、第一次尝试

1、实现目标：

实现一个程序，以验证码图片作为输入，通过某种识别算法，输出识别后的结果。

2、解决思路

首先必须对上一步得到的图片人工标记结果，标记的结果为'A'-'Z' '0'-'9'中的一个，之后输入训练集中的图片，每个图片是 12*14 的黑白图片，将其展开为 168 维的 0-1 向量，对于每个标签根据训练集求解这个标签下对应的所有图片在 168 维空间上的几何中心，即完成训练过程。

之后每输入一张图片，将其分离成四个字符图片，对四个字符图片求解其距离每一个标签的几何中心的距离，距离最近者作为识别结果。

3、程序设计

参见 knn.py。

4、运行结果

参见 result(first try).txt

**********Report**********

Total images: 100

Images guessed correct: 75

Total chars: 400

Chars guessed correct: 375

虽然单个字符识别率较高，但是对于验证码而言四个字符只要有一个识别不正确整个图片即识别错误，因此一个验证码图片识别准确率只有 75%，还需要提升。

六、第二次尝试

有没有什么方法可以进一步提升准确性呢？观察发现验证码字符中每个字母和数字都有两种，粗体和非粗体，一些字母和数字两者在空间结构上有显著区别例如 UV38 等，会干扰中心的计算，造成部分图片判断失误。因此，如果对标签进行细分，将粗体和非粗体视为两种不同字符，就可以提升准确率。

运行结果：

**********Report**********

Total images: 100

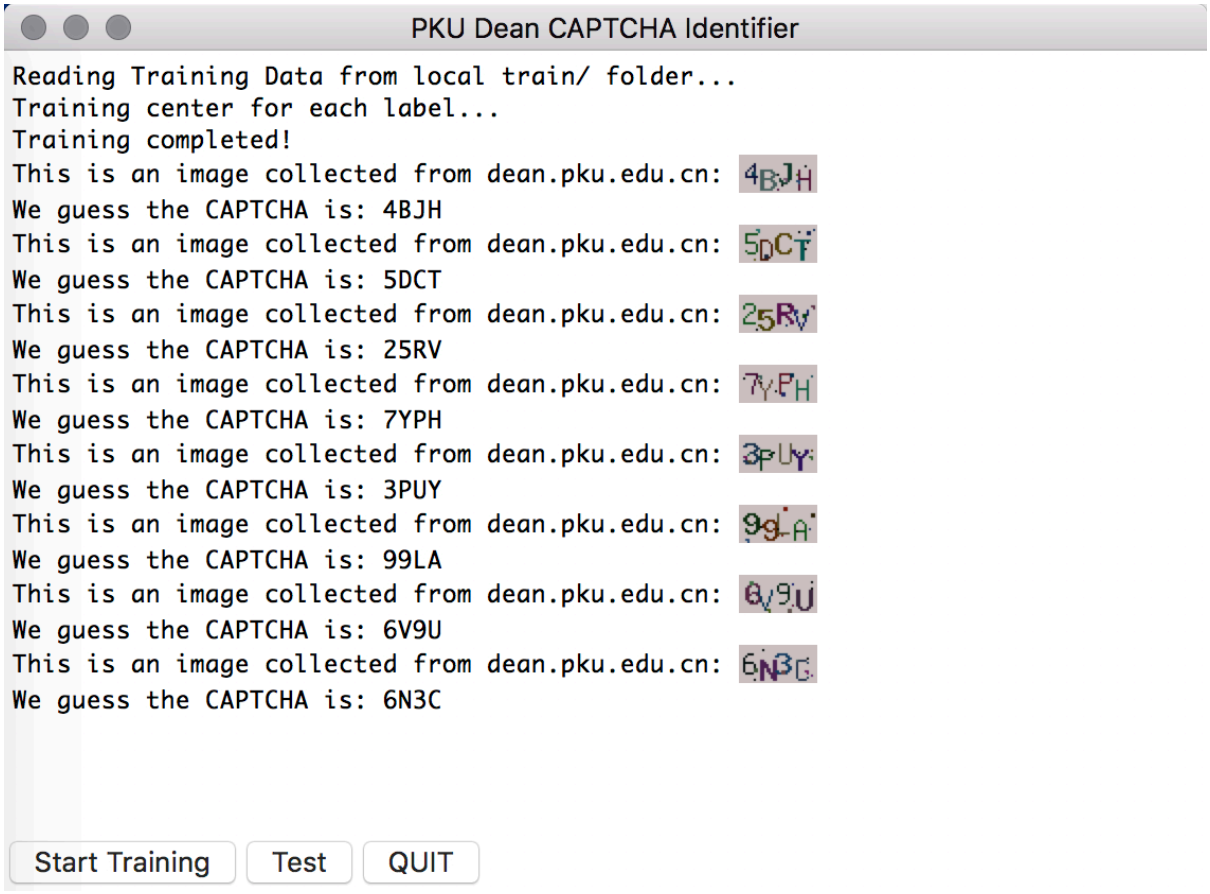
Images guessed correct: 93

Total chars: 400

Chars guessed correct: 393

整体准确率达到 93%，单个字符识别准确率为 98.25%。

下面是一个简单的 GUI 可交互的版本（参见 knn-gui.py）



七、心得体会

我们的实现的优缺点

优点：实现简单，不依赖高级机器学习库，可以很容易在不同设备和环境下实现。

缺点：泛用性低，对更加复杂的验证码将难以有所用处。