**路由技术与原理程序设计**

**--python破解验证码的设计与实现**

**实**

**验**

**报**

**告**

**班 级 2016211302**

**姓 名 毛维嘉**

**学 号 2016211067**

1. 实验内容

生活中，我们在登录微博，邮箱的时候，常常会碰到验证码。在工作时，如果想要爬取一些数据，也会碰到验证码的阻碍。本次试验将带领大家认识验证码的一些特性，并利用 Python 中的 pillow 库完成对验证码的破解。

**项目要求：**

* + - 预处理，提取文本图片。

1-4.1

* + - 提取单个字符图片。
    - 使用向量空间搜索引擎或神经网络来做字符识别
    - 识别率达到95%以
    - 验证码数据见参考资料第二条

1. 实验原理方法
2. 原始图片处理

我们第一步是从对应的文件夹中取出测试图片以及相应的样例，这里面我们采用python文件读取函数进行操作

def getPic():  
 rootdir="./python\_captcha/python\_captcha/examples"  
 list = os.listdir(rootdir) # 列出文件夹下所有的目录与文件  
 img=[]  
 for i in range(0, len(list)):  
 path = os.path.join(rootdir, list[i])  
 if os.path.isfile(path) and path[len(path)-3:]=="gif":  
 im=Image.open(path)  
 img.append(im)  
 return img

我们首先获得要破解的图片，上面按照顺序排列着一些数字和字母，我们现在要做的就是识别出来对应的数字和字母。

首先我们需要将图片转化为8位像素。

那么我们接下来要做的就是颜色的处理，因为我们知道我们输入的图片很有可能是彩色的，所以这个时候我们可以做一个类似二值化的处理过程，将所有颜色变为黑白两色，这样就可以在识别和提取的时候更为容易。

def changeColor(im,image2,t):  
 for x in range(im.size[1]):  
 for y in range(im.size[0]):  
 pix = im.getpixel((y, x))  
 t[pix] = pix  
 if pix == 220 or pix == 227: # these are the numbers to get  
 image2.putpixel((y, x), 0)  
 return image2,t

然后我们要做的就是分割整个图片中的数字和字母，记录每个数字或者字符对应的起始和结束为止。这个时候我们采用的是遍历的办法来判断每个字符的边界，并将其记录下来。

def recordPos(image2):  
 letterlist=[]  
 wordin = False  
 wordexist = False  
 start = 0  
 end = 0  
 for y in range(image2.size[0]): # slice across  
 for x in range(image2.size[1]): # slice down  
 pix = image2.getpixel((y, x))  
 if pix != 255:  
 wordin = True  
  
 if wordexist == False and wordin == True:  
 wordexist = True  
 start = y  
  
 if wordexist == True and wordin == False:  
 wordexist = False  
 end = y  
 letterlist.append((start, end))  
  
 wordin = False  
 return letterlist

1. 计算相似度

这里面有我们的原始训练样例和图片

然后我们就将目前要识别的图片和原始训练样例进行相似度的匹配，这个我们采用的是向量空间来进行相似度的判定，我们通过计算向量的余弦夹角，夹角越小代表着相似度越高。

接下来我会详细介绍一下相似度如何计算。

首先我们要取得每个图片的向量表示，这个地方我们用一个1 0 向量进行表示。我们现在已经知道到了很多样例的数字以及字母的向量表示，然后我们将我们每个测试的字符或者数字的向量分别与每一个样例进行余弦相似度的计算，也就是计算两个向量的余弦夹角

Cos a= ab/|a||b|

然后我们对夹角进行排序，排序夹角最小的就是最相似的。

我们在这里定义了一个类来计算相应的相似度

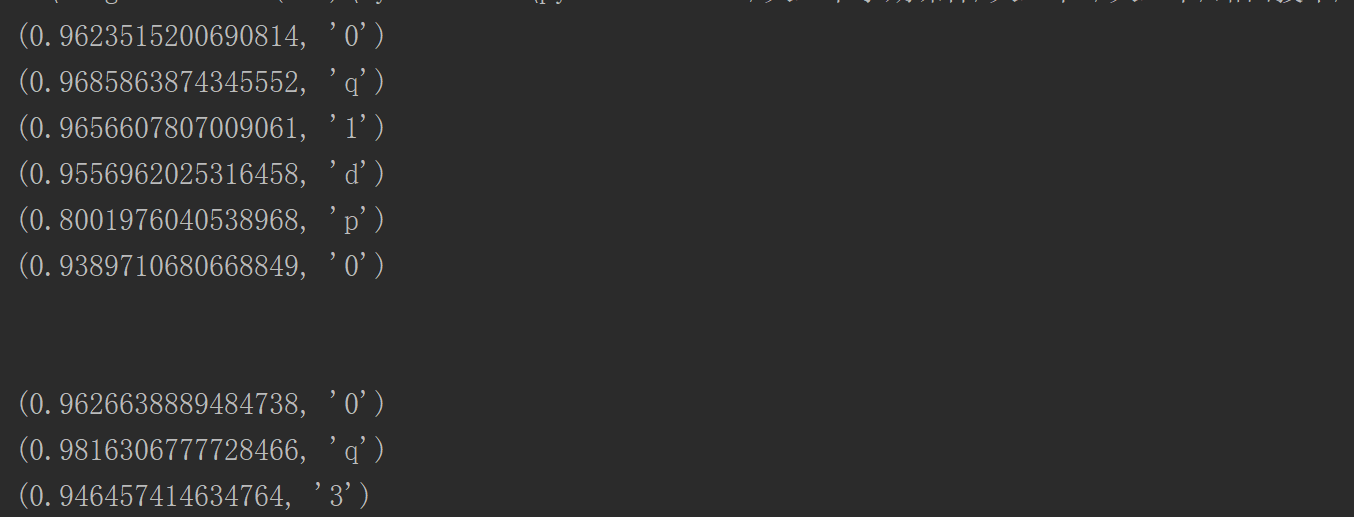
class VectorCompare:  
 def magnitude(self,concordance):  
 total = 0  
 for word,count in concordance.items():  
 total += count \*\* 2  
 return math.sqrt(total)  
  
 def relation(self,concordance1, concordance2):  
 relevance = 0  
 topvalue = 0  
 for word,count in concordance1.items():  
 #print(word)  
 if word in concordance2:  
 topvalue += count \* concordance2[word]  
 return topvalue / (self.magnitude(concordance1) \* self.magnitude(concordance2))

这个类里面两个函数用来计算每个向量的模值以及夹角大小。

class VectorCompare:  
 def magnitude(self,concordance):  
 total = 0  
 for word,count in concordance.items():  
 total += count \*\* 2  
 return math.sqrt(total)  
  
 def relation(self,concordance1, concordance2):  
 relevance = 0  
 topvalue = 0  
 for word,count in concordance1.items():  
 #print(word)  
 if word in concordance2:  
 topvalue += count \* concordance2[word]  
 return topvalue / (self.magnitude(concordance1) \* self.magnitude(concordance2))

这两个函数一个用来计算模值，一个用来计算向量点乘，两者相除即可得到相似度的大小比较。

1. 结果输出



1. 实验环境

Python 3.7

IDE：pycharm

1. 函数列表

|  |  |  |
| --- | --- | --- |
| 函数名 | 参数 | 作用 |
| Def VectorGet(im) | im | 得到相应图片的向量 |
| Def getLetter() | 无 | 得到相应的图片 |
| Def changeColor（） | Im,image2,t | 改变图片中的颜色 |
| Def recordPos（） | Image2 | 记录每个字母数字的位置 |
| Def guessLetter() | Letterlist image2 | 进行相似度比较并给出预测 |
| Def main() | 无 | 主函数 |

1. 实验心得以及感悟

完成了本次实验之后，我对Python有了更加深刻的理解和体会，学会了很多有趣的知识和应用。对验证码的识别有了一些理解，也明白了如何使用向量空间余弦夹角的相似度来进行匹配。这个技术不仅可以用在验证码的识别上面，也可以用在例如 nlp 模式匹配等诸多领域，有着非常广泛的应用，今天对这个有所理解和掌握也算是一种很大的进步，让我对很多有趣的实际应用有了更加深刻的理解。

1. 源代码
2. from PIL import Image  
   import hashlib  
   import time  
   import os  
   import math  
     
   iconset = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '0', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j',  
    'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']  
   imageset = []  
     
   class VectorCompare:  
    def magnitude(self,concordance):  
    total = 0  
    for word,count in concordance.items():  
    total += count \*\* 2  
    return math.sqrt(total)  
     
    def relation(self,concordance1, concordance2):  
    relevance = 0  
    topvalue = 0  
    for word,count in concordance1.items():  
    #print(word)  
    if word in concordance2:  
    topvalue += count \* concordance2[word]  
    return topvalue / (self.magnitude(concordance1) \* self.magnitude(concordance2))  
     
   v = VectorCompare()  
     
   def VectorGet(im):  
    vector = {}  
    count = 0  
    for i in im.getdata():  
    vector[count] = i  
    count += 1  
    return vector  
     
   def getLetter():  
    for letter in iconset:  
    for img in os.listdir('python\_captcha/python\_captcha/iconset/%s/' % (letter)):  
    t = []  
    if img != "Thumbs.db" and img != ".DS\_Store": # windows check...  
    t.append(VectorGet(Image.open("python\_captcha/python\_captcha/iconset/%s/%s" % (letter, img))))  
    imageset.append({letter: t})  
     
   def changeColor(im,image2,t):  
    for x in range(im.size[1]):  
    for y in range(im.size[0]):  
    pix = im.getpixel((y, x))  
    t[pix] = pix  
    if pix == 220 or pix == 227: # these are the numbers to get  
    image2.putpixel((y, x), 0)  
    return image2,t  
     
     
   def recordPos(image2):  
    letterlist=[]  
    wordin = False  
    wordexist = False  
    start = 0  
    end = 0  
    for y in range(image2.size[0]): # slice across  
    for x in range(image2.size[1]): # slice down  
    pix = image2.getpixel((y, x))  
    if pix != 255:  
    wordin = True  
     
    if wordexist == False and wordin == True:  
    wordexist = True  
    start = y  
     
    if wordexist == True and wordin == False:  
    wordexist = False  
    end = y  
    letterlist.append((start, end))  
     
    wordin = False  
    return letterlist  
     
   def guessLetter(letterlist,image2):  
    count = 0  
    for letter in letterlist:  
    m = hashlib.md5()  
    image3 = image2.crop((letter[0], 0, letter[1], image2.size[1]))  
    predict = []  
     
    for image in imageset:  
    for x, y in image.items():  
    if len(y) != 0:  
    predict.append((v.relation(y[0], VectorGet(image3)), x))  
     
    predict.sort(reverse=True)  
    print("", predict[0])  
    count += 1  
     
   def getPic():  
    rootdir="./python\_captcha/python\_captcha/examples"  
    list = os.listdir(rootdir) # 列出文件夹下所有的目录与文件  
    img=[]  
    for i in range(0, len(list)):  
    path = os.path.join(rootdir, list[i])  
    if os.path.isfile(path) and path[len(path)-3:]=="gif":  
    im=Image.open(path)  
    img.append(im)  
    return img  
     
   def main ():  
    getPic()  
    getLetter()  
    img=getPic()  
    #im = Image.open("python\_captcha/python\_captcha/captcha.gif")  
    for im in img:  
    image2 = Image.new("P", im.size, 255)  
    im.convert("P")  
    t = {}  
    image2,t=changeColor(im,image2,t)  
    letterlist=recordPos(image2)  
    guessLetter(letterlist,image2)  
    print("\n")  
     
     
   if \_\_name\_\_ == '\_\_main\_\_':  
    main()