**Python 破解验证码**

**一、实验说明**

**1. 环境登录**

无需密码自动登录，系统用户名shiyanlou

**2. 环境介绍**

本实验环境采用带桌面的Ubuntu Linux环境，实验中会用到桌面上的程序：

1. LX终端（LXTerminal）: Linux命令行终端，打开后会进入Bash环境，可以使用Linux命令
2. GVim：非常好用的编辑器，最简单的用法可以参考课程[Vim编辑器](http://www.shiyanlou.com/courses/2)

**3. 环境使用**

使用GVim编辑器输入实验所需的代码及文件，使用LX终端（LXTerminal）运行所需命令进行操作。

实验报告可以在个人主页中查看，其中含有每次实验的截图及笔记，以及每次实验的有效学习时间（指的是在实验桌面内操作的时间，如果没有操作，系统会记录为发呆时间）。这些都是您学习的真实性证明。

**4. 知识点**

本节实验中将学习和实践以下知识点：

1. Python基本知识
2. PIL模块的使用

**二、实验内容**

本实验将通过一个简单的例子来讲解破解验证码的原理。

安装 pillow（PIL）库：

$ sudo apt-get update

$ sudo apt-get install python-dev

$ sudo apt-get install libtiff5-dev libjpeg8-dev zlib1g-dev \

libfreetype6-dev liblcms2-dev libwebp-dev tcl8.6-dev tk8.6-dev python-tk

$ sudo pip install pillow

下载实验用的文件：

$ wget http://labfile.oss.aliyuncs.com/courses/364/python\_captcha.zip

$ unzip python\_captcha.zip

$ cd python\_captcha

这是我们实验使用的验证码 captcha.gif

处输入图片的描述

**提取文本图片**

在工作目录下新建 crack.py 文件，进行编辑。

#-\*- coding:utf8 -\*-

from PIL import Image

im = Image.open("captcha.gif")

#(将图片转换为8位像素模式)

im = im.convert("P")

#打印颜色直方图

print im.histogram()

输出：

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 2, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 1, 0, 0, 0, 2, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0 , 1, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 0, 0, 1, 2, 0, 1, 0, 0, 1, 0, 2, 0, 0, 1, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 3, 1, 3, 3, 0, 0, 0, 0, 0, 0, 1, 0, 3, 2, 132, 1, 1, 0, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 15, 0 , 1, 0, 1, 0, 0, 8, 1, 0, 0, 0, 0, 1, 6, 0, 2, 0, 0, 0, 0, 18, 1, 1, 1, 1, 1, 2, 365, 115, 0, 1, 0, 0, 0, 135, 186, 0, 0, 1, 0, 0, 0, 116, 3, 0, 0, 0, 0, 0, 21, 1, 1, 0, 0, 0, 2, 10, 2, 0, 0, 0, 0, 2, 10, 0, 0, 0, 0, 1, 0, 625]

颜色直方图的每一位数字都代表了在图片中含有对应位的颜色的像素的数量。

每个像素点可表现256种颜色，你会发现白点是最多（白色序号255的位置，也就是最后一位，可以看到，有625个白色像素）。红像素在序号200左右，我们可以通过排序，得到有用的颜色。

his = im.histogram()

values = {}

for i in range(256):

values[i] = his[i]

for j,k in sorted(values.items(),key=lambda x:x[1],reverse = True)[:10]:

print j,k

输出：

255 625

212 365

220 186

219 135

169 132

227 116

213 115

234 21

205 18

184 15

我们得到了图片中最多的10种颜色，其中 220 与 227 才是我们需要的红色和灰色，可以通过这一讯息构造一种黑白二值图片。

#-\*- coding:utf8 -\*-

from PIL import Image

im = Image.open("captcha.gif")

im = im.convert("P")

im2 = Image.new("P",im.size,255)

for x in range(im.size[1]):

for y in range(im.size[0]):

pix = im.getpixel((y,x))

if pix == 220 or pix == 227: # these are the numbers to get

im2.putpixel((y,x),0)

im2.show()

得到的结果：

处输入图片的描述

**提取单个字符图片**

接下来的工作是要得到单个字符的像素集合，由于例子比较简单，我们对其进行纵向切割：

inletter = False

foundletter=False

start = 0

end = 0

letters = []

for y in range(im2.size[0]):

for x in range(im2.size[1]):

pix = im2.getpixel((y,x))

if pix != 255:

inletter = True

if foundletter == False and inletter == True:

foundletter = True

start = y

if foundletter == True and inletter == False:

foundletter = False

end = y

letters.append((start,end))

inletter=False

print letters

输出：

[(6, 14), (15, 25), (27, 35), (37, 46), (48, 56), (57, 67)]

得到每个字符开始和结束的列序号。

import hashlib

import time

count = 0

for letter in letters:

m = hashlib.md5()

im3 = im2.crop(( letter[0] , 0, letter[1],im2.size[1] ))

m.update("%s%s"%(time.time(),count))

im3.save("./%s.gif"%(m.hexdigest()))

count += 1

(接上面的代码)

对图片进行切割，得到每个字符所在的那部分图片。

**AI 与向量空间图像识别**

在这里我们使用向量空间搜索引擎来做字符识别，它具有很多优点：

* 不需要大量的训练迭代
* 不会训练过度
* 你可以随时加入／移除错误的数据查看效果
* 很容易理解和编写成代码
* 提供分级结果，你可以查看最接近的多个匹配
* 对于无法识别的东西只要加入到搜索引擎中，马上就能识别了。

当然它也有缺点，例如分类的速度比神经网络慢很多，它不能找到自己的方法解决问题等等。

关于向量空间搜索引擎的原理可以参考这篇文章：<http://ondoc.logand.com/d/2697/pdf>

Don't panic。向量空间搜索引擎名字听上去很高大上其实原理很简单。拿文章里的例子来说：

你有 3 篇文档，我们要怎么计算它们之间的相似度呢？2 篇文档所使用的相同的单词越多，那这两篇文章就越相似！但是这单词太多怎么办，就由我们来选择几个关键单词，选择的单词又被称作特征，每一个特征就好比空间中的一个维度（x，y，z 等），一组特征就是一个矢量，每一个文档我们都能得到这么一个矢量，只要计算矢量之间的夹角就能得到文章的相似度了。

用 Python 类实现向量空间：

import math

class VectorCompare:

#计算矢量大小

def magnitude(self,concordance):

total = 0

for word,count in concordance.iteritems():

total += count \*\* 2

return math.sqrt(total)

#计算矢量之间的 cos 值

def relation(self,concordance1, concordance2):

relevance = 0

topvalue = 0

for word, count in concordance1.iteritems():

if concordance2.has\_key(word):

topvalue += count \* concordance2[word]

return topvalue / (self.magnitude(concordance1) \* self.magnitude(concordance2))

它会比较两个 python 字典类型并输出它们的相似度（用 0～1 的数字表示）

**将之前的内容放在一起**

还有取大量验证码提取单个字符图片作为训练集合的工作，但只要是有好好读上文的同学就一定知道这些工作要怎么做，在这里就略去了。可以直接使用提供的训练集合来进行下面的操作。

iconset目录下放的是我们的训练集。

最后追加的内容：

#将图片转换为矢量

def buildvector(im):

d1 = {}

count = 0

for i in im.getdata():

d1[count] = i

count += 1

return d1

v = VectorCompare()

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']

#加载训练集

imageset = []

for letter in iconset:

for img in os.listdir('./iconset/%s/'%(letter)):

temp = []

if img != "Thumbs.db" and img != ".DS\_Store":

temp.append(buildvector(Image.open("./iconset/%s/%s"%(letter,img))))

imageset.append({letter:temp})

count = 0

#对验证码图片进行切割

for letter in letters:

m = hashlib.md5()

im3 = im2.crop(( letter[0] , 0, letter[1],im2.size[1] ))

guess = []

#将切割得到的验证码小片段与每个训练片段进行比较

for image in imageset:

for x,y in image.iteritems():

if len(y) != 0:

guess.append( ( v.relation(y[0],buildvector(im3)),x) )

guess.sort(reverse=True)

print "",guess[0]

count += 1

全部代码：

from PIL import Image

import hashlib

import time

import os

import math

class VectorCompare:

def magnitude(self,concordance):

total = 0

for word,count in concordance.iteritems():

total += count \*\* 2

return math.sqrt(total)

def relation(self,concordance1, concordance2):

relevance = 0

topvalue = 0

for word, count in concordance1.iteritems():

if concordance2.has\_key(word):

topvalue += count \* concordance2[word]

return topvalue / (self.magnitude(concordance1) \* self.magnitude(concordance2))

def buildvector(im):

d1 = {}

count = 0

for i in im.getdata():

d1[count] = i

count += 1

return d1

v = VectorCompare()

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 = []

for letter in iconset:

for img in os.listdir('./iconset/%s/'%(letter)):

temp = []

if img != "Thumbs.db" and img != ".DS\_Store": # windows check...

temp.append(buildvector(Image.open("./iconset/%s/%s"%(letter,img))))

imageset.append({letter:temp})

im = Image.open("captcha.gif")

im2 = Image.new("P",im.size,255)

im = im.convert("P")

temp = {}

for x in range(im.size[1]):

for y in range(im.size[0]):

pix = im.getpixel((y,x))

temp[pix] = pix

if pix == 220 or pix == 227: # these are the numbers to get

im2.putpixel((y,x),0)

inletter = False

foundletter=False

start = 0

end = 0

letters = []

for y in range(im2.size[0]): # slice across

for x in range(im2.size[1]): # slice down

pix = im2.getpixel((y,x))

if pix != 255:

inletter = True

if foundletter == False and inletter == True:

foundletter = True

start = y

if foundletter == True and inletter == False:

foundletter = False

end = y

letters.append((start,end))

inletter=False

count = 0

for letter in letters:

m = hashlib.md5()

im3 = im2.crop(( letter[0] , 0, letter[1],im2.size[1] ))

guess = []

for image in imageset:

for x,y in image.iteritems():

if len(y) != 0:

guess.append( ( v.relation(y[0],buildvector(im3)),x) )

guess.sort(reverse=True)

print "",guess[0]

count += 1

**得到结果**

一切准备就绪，运行我们的代码试试：

python crack.py

输出

(0.96376811594202894, '7')

(0.96234028545977002, 's')

(0.9286884286888929, '9')

(0.98350370609844473, 't')

(0.96751165072506273, '9')

(0.96989711688772628, 'j')

是正解，干得漂亮。