“希希敬敬对”团队项目代码及界面展示

1、百度贴吧爬虫小程序项目代码

（1）GUI.py程序开始

#程序开始界面

from tkinter import \*

from WebCrawler.Start\_search import Start

root = Tk()

root.title("百度贴吧数据小爬虫")

root.geometry('1000x600') # 是x 不是\*

root.resizable(0,0)

img\_gif = PhotoImage(file='web-crawler.gif')

label\_img = Label(root, image=img\_gif)

label\_img.place(relx=0.24, rely=0.25)

var = StringVar()

label\_info = Label(root, text="请输入要爬取的贴吧名称或URL地址",

font=("Arial", 12), fg='#3385ff',relief=FLAT )

label\_info.place(relx=0.24, rely=0.5)

entry1 = Entry(root, width=40, font=("Arial", 16), fg='grey', relief=FLAT)

entry1.place(relx=0.24, rely=0.55)

label\_wait = Label(root, text='点击"开始爬取"后请耐心等待10秒。。。',

font=("Arial", 12), fg='#000020', relief=FLAT)

label\_wait.place(relx=0.24, rely=0.65)

def button\_click():

text = entry1.get()

start = Start()

start.start\_Crawler(text)

button = Button(root, text="开始爬取", width=10, height=1,

bg='#3385ff', fg='WHITE', relief=FLAT, command=lambda : button\_click())

button.place(relx=0.7, rely=0.55)

root.mainloop()

（2）Start\_search.py程序

from WebCrawler import UrlList

from WebCrawler.Crawler import BDTBCrawler

from WebCrawler.draw import Creat\_form

#from urllib import parse

class Start():

def start\_Crawler(self, text):

#调用UrlList类

my\_url = UrlList.Url()

url = my\_url.creat\_url(text)

#调用爬虫类开始爬取

crawler = BDTBCrawler(url)

anchors = crawler.go()

anchors = anchors[0:10]

#调用画图页面输出结果

draw\_form = Creat\_form(text, anchors)

for i in anchors:

print(i)

（3）UrlList.py程序

'''

使用list将百度贴吧中贴吧名称与URL链接对应

'''

import re

from urllib.request import quote

import tkinter.messagebox

class Url():

def creat\_url(self,text):

# print("请输入你要爬取的贴吧名称：")

# string = input()

string = text

if string == '':

tkinter.messagebox.showinfo('提示信息', '请输入正确的贴吧名称！')

return None

flag = re.findall('[(http)(HTTP)]',string)

if flag:

url = string

else:

string = quote(string)

url = "https://tieba.baidu.com/f?kw="+ string + "&ie=utf-8"

return url

（4）Crawler.py程序

# 找到数据对应的网页，分析网页结构找到数据所在的标签位置

#模拟HTTP请求，向服务器发送这个请求，获取到服务器返回给我们的HTML

#用正则表达式提取我们要的数据（主题作者、帖子回复数）

import re

from urllib import request

class BDTBCrawler():

url = "http://tieba.baidu.com/f?kw=%E4%B8%9C%E5%8D%8E%E7%90%86%E5%B7%A5%E5%A4%A7%E5%AD%A6&ie=utf-8"

Name\_num\_list = []

def \_\_init\_\_(self, url):

BDTBCrawler.url = url

#匹配到包含了主题作者和帖子回复数关键字的标签

root\_pattern = '<span class="threadlist\_rep\_numcenter\_text"([\s\S]\*?)data-field='

#匹配到对应的帖子回复数

num\_pattern = 'title="回复">([\s\S]\*?)</span>'

# 匹配到主题作者

name\_pattern = 'title="主题作者: ([\s\S]\*?)"'

#模拟HTTP请求，向服务器发送请求，获取到服务器返回给我们的HTML

def \_\_fetch\_content(self):

r = request.urlopen(BDTBCrawler.url)

htmls = r.read()

# 将服务器返回的字节码转换成字符串格式

htmls = str(htmls, encoding='utf-8')

return htmls

#分析获取到的字符串

def \_\_analysis(self, htmls):

#root\_html获取包含了主题作者和帖子回复数关键字的标签

root\_html = re.findall(BDTBCrawler.root\_pattern, htmls)

#用anchors这个列表来存放提取出来的主题作者和帖子回复数组成的字典

anchors = []

for html in root\_html:

# 提取主题作者（列表类型），并将其转换成字符串

name = re.findall(BDTBCrawler.name\_pattern, html)

name = str(name[0])

# #提取回复数（列表类型），并将其转换成整形数字

number = re.findall(BDTBCrawler.num\_pattern, html)

number = int(number[0])

#用来记录列表的遍历过程中其子元素——字典中是否有与当前name相同的key值

flag = False

# 遍历anchors列表，如果有相同的主题作者执行回复数累加操作

for i in anchors:

if name == i['name']:

number = i['number'] + number

# print(number)

i.update({'name':i['name'], 'number':number})

flag = True #有与当前作者相同的主题作者

break

if flag == False:

anchor = {'name': name, 'number': number}

anchors.append(anchor)

# print(anchors)

return anchors

#排序算法

def \_\_sort(self, anchors):

anchors = sorted(anchors, key=lambda d: d['number'], reverse=True)

return anchors

def go(self):

#使用for循环爬取前10页

htmls = ''

for i in range(0, 10):

pn = i \* 50

#page记录当前爬取页面需要在URL上添加的字符串

page = '&pn=' + str(pn)

BDTBCrawler.url += page

htmls += self.\_\_fetch\_content()

anchors = self.\_\_analysis(htmls)

anchors = self.\_\_sort(anchors)

for i in anchors:

print(i)

return anchors

（5）draw.py程序

import matplotlib.pyplot as plt

import matplotlib

import numpy as np

from tkinter import \*

from matplotlib.pylab import mpl

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg #NavigationToolbar2TkAgg

from matplotlib.figure import Figure

class Creat\_form():

def \_\_init\_\_(self, school, anchors):

self.root = Tk() #创建主窗体

self.root.title("百度贴吧数据小爬虫")

self.root.geometry('1000x600') # 是x 不是\*

self.root.resizable(0, 0)

self.canvas = Canvas() #创建一块显示图形的画布

self.figure = self.create\_matplotlib(school, anchors) #返回matplotlib所画图形的figure对象

self.create\_form(self.figure) #将figure显示在tkinter窗体上面

self.root.mainloop()

def create\_matplotlib(self, school, anchors):

# 设置中文字体正常显示

plt.rcParams['font.sans-serif'] = ['SimHei']

# 设置负号正常显示

plt.rcParams['axes.unicode\_minus'] = False

f = plt.figure(num = 2, figsize=(10,6),dpi=80, frameon = True) #创建绘图对象f

fig1 = plt.subplot(1,1,1) #创建一个子图

font1 = {'weight': 'normal',

'size': 18,

}

# 条形图标题设置

plt.title(school + "贴吧前10页主题作者发帖回复总数排行榜",font1)

# 设置X，Y轴标签

font2 = {'weight': 'normal',

'size': 14,

}

plt.xlabel("主题作者",font2)

plt.ylabel("回复总数",font2)

# 横、纵坐标刻度显示值

row\_list = []

col\_list = []

for kv in anchors:

row\_list.append(kv['name'])

col\_list.append(kv['number'])

# print(row\_list)

# print(col\_list)

# row\_list = ['七七', '火星', 'later', 'maybe','like','science']

# col\_list = [15, 20, 30, 45, 60, 70]

# 绘制条形图

# height：长条形的高度

# width:长条形宽度，默认值0.8

x = range(len(col\_list))

rects = plt.bar(x, height=col\_list, width=0.2, alpha=0.8, color='blue')

# 设置y轴的取值范围

plt.ylim(0,col\_list[0] + 1000)

# 设置X轴刻度显示值（参数一：中点坐标，参数二：显示值）

plt.tick\_params(labelsize = 12)

plt.xticks([index for index in x], row\_list,rotation = 40)

#绘制条形图上的数据标签

for rect in rects:

height = rect.get\_height()

plt.text(rect.get\_x() + rect.get\_width() / 2, height + 1 ,

str(height), ha = "center", va = "bottom")

plt.show()

return f

def create\_form(self, figure):

#把绘制的图形显示到tkinter窗口上

self.canvas = FigureCanvasTkAgg(figure, self.root)

self.canvas.draw()

self.canvas.get\_tk\_widget().place(relx = 0.10,rely = 0.10)

2、百度贴吧爬虫小程序项目界面展示



