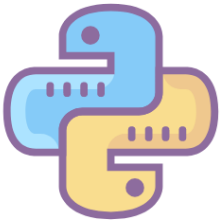

CS 104 : Python based web-crawler project

Busa Siva Naga Venkata Aditya[22B1024]

Department of Computer Science

Indian Institute of Technology, BOMBAY

June 2023

1 Summary

In this report, we will explore a web crawling code that extracts information from a website using Python and the BeautifulSoup library. The code aims to scrape data from a specific webpage and store it in a text file or print it in the terminal

2 Necessary libraries and their Functions

```
import requests :

    Gets source code from the URL mentioned

import argparse :

    It provides a mechanism for parsing command-line arguments and option

from urllib.parse import urlparse, urljoin :

    The urljoin function is used to resolve a relative URL against a base
    URL and return the absolute URL.
    The urlparse function is used to parse a URL string and break it down
    into its individual components. It returns a named tuple containing
    the scheme, netloc, path, parameters, query, and fragment of the URL

from bs4 import BeautifulSoup :

    It is used for web scraping and parsing HTML and XML documents. It
    provides a convenient way to extract data from HTML or XML files by
    creating a parse tree that can be navigated using methods and filters.

import urllib3 :

    is a powerful HTTP client library for Python. It provides functionality
    for making HTTP requests, handling connection pooling, SSL/TLS
    verification, and various other features related to HTTP communication

import os :
    It provides a way to interact with the operating system. It offers
    functions and methods for various operating system-related tasks,
    including file and directory operations, process management,
    environment variables, and more
    Here, in this code, we use this to extract extension of the URL
```

3 User-Defined Functions

There are several User-defined Functions in the code, this section gives a basic idea on how the Functions are constructed and implemented

3.1 get_file_extension :

Defined at : Line 8

This function takes in the URL and returns the file extension of it.

3.2 get_internal_links :

Defined at : Line 326

This function takes in the url, accesses it using requests and BeautifulSoup, finds all possible tags which can have href and src as attribute and from them it takes out the links and return it as a set.

3.3 crawl :

Defined at : Line 380

For a given URL, it obtains the links in it using **get_internal_links** function and use **urljoin** to transform relative links into absolute links and it uses **urlparse** to get the domain of the link which is required since we have to crawl only Internal Links.

3.4 web_crawler :

Defined at : Line 362

This is the major function around which our whole Code runs, This function utilises every other function,it takes in our all inputs from command line and processes them using all the above functions.It even prints the output.

3.5 main :

Defined at : Line 512

This function is for adding arguments/options to command-line.

3.5.1 Arguments

-u : Website URL
-t : Recursion Threshold (The URL is crawled in it's entirety if -t is not mentioned)
-c : Type of Customisation (0 for Default)
-o : File to save the output(If the file doesn't exist then the code creates a file with it's name at the same hierarchy level

4 Customisations

4.1 custom=0

It is the default customisation as described by the question

It contains :

```
At recursion level : 1
Total files found are : 191

Type of File = JPEG Image(.jpg)(No. of files = 8) :
https://www.iitb.ac.in/sites/www.iitb.ac.in/files/reshigh/2022-08/pic.jpg
https://www.iitb.ac.in/sites/www.iitb.ac.in/files/reshigh/2023-06/img.jpg
https://www.iitb.ac.in/sites/www.iitb.ac.in/files/reshigh/2022-09/image1.jpg
https://www.iitb.ac.in/sites/www.iitb.ac.in/files/reshigh/2022-09/img1.jpg
https://www.iitb.ac.in/sites/www.iitb.ac.in/files/reshigh/2023-02/img1.jpg
https://www.iitb.ac.in/sites/www.iitb.ac.in/files/reshigh/2023-04/pic.jpg
https://www.iitb.ac.in/sites/www.iitb.ac.in/files/reshigh/2022-08/img17.jpg
https://www.iitb.ac.in/sites/www.iitb.ac.in/files/reshigh/2023-02/img.jpg
```

- **Depth level :**

  The first line shows the Recursion Threshold given to it by user on the command line

- **Total count :**

  The second line shows the total number of unique links printed in the output

- **Type of file and No.of Files :**

  This is the 4th line here,it gives the type of file based on the extension of URL and the no.of such files

- **Links :**

  Finally, These are the links we have successfully crawled, **Hurray!**

4.2 custom=1

This customisation just prints all the links together without any segregation.

4.3 custom=2

This customisation prints Internal and External Links separately.
This is executed by checking the domain of every link using **urlparse** function

```
1    At recursion level : 1
2    Total files found are : 191
3
4    External links :
5    https://asc.iitb.ac.in/acadmenu/
```

```
50   https://ep.iitb.ac.in/jobsearch
51   Internal links :
52   http://www.iitb.ac.in/calendar-node-field-event-date
53   http://www.iitb.ac.in/safety/
54   https://www.iitb.ac.in/safety/
```

4.4 custom=3

This customisation prints links Depth-wise(i.e if we give ***Threshold value = 3*** then the code executes and finds out what links and obtained at ***depth = 1***, print them and then it prints unique links found at ***depth = 2*** and then same for ***depth = 3***.

```
1    At recursion level : 1
2    Total files found are : 191
3
4
5    New Links found at depth level 1 :
6
7    https://www.iitb.ac.in/newacadhome/toadmission.jsp
```

This is achieved by making a dictionary with keys as depth and values as empty sets to which we add the links as there being recursively crawled and later while printing/saving the output, we would uniquely print the links

5 Error Handling and Changes made from the Base Code

5.1 urllib3.disable_warnings

First Occurrence : Line 15

If we verify SSL certificate for every link then some websites like *http://www.iitb.ac.in* do not give a result,as we are just crawling and printing the resulting links,we don't have to verify certificate but if we don't verify it,some warnings are floated,to neglect them we use the **urllib3.disable_warnings** function.

5.2 Redirecting links

requests.get() automatically handles redirection and doesn't require us to explicitly add *allow_redirects*

5.3 Handling internal referencing

The below code snippet handles links with # ,which are included in fragment section of **urlparse**. These links are not unique webpages rather they just reference a part of the same webpage

```
if link and not link.startswith('#'): #To remove fragment
    while link.startswith(' '): # to remove some unnecessa
        link=link[1:]
    parse_url=urlparse(link) #To remove fragment identifie
    fragment_length=len(parse_url.fragment)
    link_length=len(link)
    if fragment_length!=0:
        unique_link=link[0:link_length-fragment_length-1]
    else:
        unique_link=link
    internal_links.add(unique_link)
```

[1] [2][3][4][5]

References

[1] URL: https://chat.openai.com/.

[2] URL: https://docs.python.org/3/library/warnings.html.

[3] URL: https://stackoverflow.com/questions/541390/extracting-extension-from-filename-in-python.

[4] URL: https://www.w3schools.com/python/python_try_except.asp.

[5] URL: https://stackoverflow.com/questions/68613912/web-scraping-troublesome-characters-could-not-be-decoded-and-were-replaced-w.