University of Groningen

Web Scraping Tool

Short Programming Project

Final Report

Version: 1.0

Author:
Keiko Angela Nicolasky(S3807452)

Supervisor:
Estefanía Talavera Martinez

Contents

1 Introduction

Web scraping is a term referring to data extraction from websites, usually done automatically by a bot, although it can be done manually by a user. Web scraping is increasingly being used, as it is especially useful for price monitoring, market research, news monitoring, sentiment analysis, and email marketing. Web scraping tools differ from each other in term of functionality and features, depending on the websites the tool is trying to scrape data from, as there are various types of websites. However, web scraping tools generally work in a similar way, which involves accessing the data and then extracting the data.

The goal for this project is to create a user-friendly web scraping tool for the purpose of dataset building. The main websites or social media platforms that the tool can scrape are Google Image, Instagram, and Twitter. These three websites have different APIs, requiring the tool to be split into different functionalities and features. The main type of data that the tool is expected to scrape are images, texts and their metadata. However, videos will also be scraped by Instagram scraper.

I was expected to create the web scraping tool for Google Image, Instagram, and Twitter as user-friendly as possible, along with a user guide, such as users will be able to enter keywords and receive data relating to those keywords. The data scraped are expected to have no duplicates. As for both Instagram and Twitter, I was expected to make an extra feature that let users enter usernames and receive posts related to those usernames. For Instagram, I was expected to make another extra feature that let users search for posts that contain two specific hashtags.

This report begins with an explanation of tools used to develop the web scraping tool, followed by list of features and functionalities that I have implemented. An explanation of those features and functionalities will also be incorporated. Finally, recommendations for future work and conclusions are discussed.

2 Tools

The web scraping tool was developed with Python 3.8, however Python 3.6 - Python 3.9 should be supported with the tool, as there are several Python packages that only works on specific range of Python versions. Beside having a correct version Python installed, there are also some packages that are required to be installed. Those packages, along with their versions are listed in `requirements.txt` that user have access to when downloading the code from GitHub repositories. An explanation for downloading the packages is available in the user guide.

3 Software Functionalities

This section describes software functionalities for each platforms.

3.1 Google Image

Google Image scraper main functionalities are:

- Download images from Google Image, without duplicates.
- Download metadata of image into a `csv` file, which describes the caption, width, and height of each image.

The way Google Image scraper works is a bit different from Instagram scraper and Twitter scraper. Below is the general explanation of `Google_Scraper` class in `google_scraper.py` file.

In `parse_page` function, a web driver is activated, the query URL of Google Image page with the keyword is entered, and the page is scrolled until it reach the end of Google Image page. Every time the page is scrolled down, the program is paused for 0.3 seconds to prevent Google from detecting the program as a bot. After that, in `save_images` function, the program iterates through every parsed image data from the `parse_page` function, gets the relevant metadata, and saves it to a `google_log_keyword` text file. Finally, in `download_images` function, the program accesses the `google_log_keyword` file which contains the URI to each saved images, and tries to download the every images. Not every download is successful, therefore the total downloaded image and the index of the last downloaded image is written in `google_d_log_keyword`, which also helps to prevent downloading duplicate images.

3.2 Instagram

Instagram scraper main functionalities are:

- Download images, videos, or both images and videos from Instagram posts based on a hashtag without duplicates, along with their captions, with an option to filter the posts in a certain time period.
- Download images, videos, or both images and videos from Instagram posts that contain two specific hashtags without duplicates, along with their captions, with an option to filter the posts in a certain time period.

- Download images, videos, or both images and videos from an Instagram user without duplicates, along with their captions, with an option to filter the posts in a certain time period.

I use Python package `instaloader` to handle Instagram API. Below is the general explanation of `Instagram_Scraper` class in `instagram_scraper.py` file.

The function names are self explanatory. `download_post_with_one_hashtag` function is called when downloading posts based on a single hashtag, while `download_post_with_two_hashtag` function is called when downloading posts that contain two specific hashtags. Both functions are similar, the only difference is, in `download_post_with_two_hashtag` function the second hashtag is set as a filter condition while searching for posts with the first hashtag. Finally, `download_post_from_user` is called when downloading posts from an Instagram user. If user wants to download posts in a certain time period, there exists a filter in each function to filter the posts. However, if the user doesn't want to download posts in certain time period, then the most recent posts are downloaded. `instaloader` handles the problem of downloading duplicate posts, as it checks if the contents of the posts already exist in our directories.

As a side note, Instagram is very strict with automated web crawlers/bots. The access to Instagram API can be easily suspended for a period of time if user tries to download too much posts in a short period of time. As a partial solution to this, I put the program to sleep after iterating every post and log in to an Instagram account (@webscrapingtool.rug) that I made specifically for this project via `instaloader` to minimize the chance of getting suspended. It doesn't solve the problem completely, however with `instaloader`, the program will keep trying to download in `n` minutes after being suspended.

3.3 Twitter

Twitter scraper main functionalities are:

- Download tweets into `txt` files based on a keyword, along with image(s) that are included in the tweets.
- Download tweets into `txt` files from a Twitter user, along with image(s) that are included in the tweets, with an option to filter tweets in certain time period.

I use python package `tweepy` to handle Twitter API. However, to directly access Twitter API itself through `tweepy`, the program needs API key, API secret key, access token key, and access token secret key, which I requested to Twitter. Below is the general explanation of `Twitter_Scraper` class in `twitter_scraper.py` file.

`download_tweets_from_keyword` function is called when downloading tweets with a single keyword. This function gets tweets related to the keyword through `tweepy`, and tried to download each of them until the amount of tweets the user want is reached. `download_tweets_from_user` function is called when downloading tweets from a Twitter user, given that user's account is not private. This function also iterates through every tweets, optionally checking if they are in between a time period (if the user wants to download in a certain time period), and download the tweets. Both of these functions avoid downloading retweets, as retweets usually have the same content as the original tweet.

Twitter scraper is generally very fast compared to the other two scrapers, except for downloading posts in certain time period, and it may also lead to error code 429 which means that Twitter application limit is exhausted. This error might also be encountered while accessing tweets that are older than seven days from current date, because of Twitter API limitations. Unlike in Instagram scraper, user has to try running the program manually after a certain period of time has passed since getting this error.

4 User Guide

The user guide can be found in section **Annex: User Guide**.

5 Conclusion

With the initial goal of creating web scraping tool for dataset building, I conclude that the goal of this project is accomplished with the tool that scrapes images, videos, and texts from Google Image, Instagram, and Twitter. From this project, I learnt how to organize Python based project, and to utilize Python packages properly. Furthermore, I also learnt that making the tool as user-friendly as possible is an important part of building a software. For future students, I have several recommendations for each of the available scraper:

1. Google Image scraper is a bit ineffective as it saves all information about images of a keyword if a user downloads images of a keyword for the first time (the second download onward should be faster as the program doesn't have to access Google Image anymore). This is the solution I came up with to prevent downloading duplicate images, however future students may find more effective ways to download images from Google Image without duplicates.

2. As mentioned in section 3.2, Instagram is very strict when it comes to automated web crawlers/bots, therefore I would like to recommend future students to research a way to bypass Instagram API limitation. It will certainly improve the tool to download more posts in shorter period of time.

3. Currently, the option to filter tweets in certain time period is only available for downloading tweets from a Twitter user. I tried incorporating this filter for downloading tweets based on a keyword, however it took quite some times for the program to run, and eventually I got the 429 error code. Therefore, I would like to recommend future students to incorporate the option to filter tweets in certain time period for downloading tweets based on a keyword.

6 Annex: User Guide

The user guide can be found in the next page.

University of Groningen

Web Scraping Tool

Short Programming Project

User Guide

Version: 1.2

Author:
Keiko Angela Nicolasky(S3807452)

Supervisor:
Estefanía Talavera Martinez

Contents

1 Downloading the Tool

The tool is available to download from a GitHub repository. User can clone the repository from the link below:

```
https://github.com/keikoang/web-scraping-tool.git
```

This code is written in Python, and Python 3 is recommended to run it.

2 Installing Required Packages

Upon pulling the code from GitHub, user needs to install the required packages listed in `requirements.txt`. After running terminal on current working directory, user should enter the following command:

```
pip install -r requirements.txt
```

3 Entering Keywords/Usernames to be Downloaded

A `txt` file named `classes.txt` is provided when user pull the code from GitHub. In this `txt` file, user should put the keywords/usernames that he/she wants to download, separated by a newline. User can put a keyword that contain white-space character. For example 'cute ginger cat' in one line will be parsed into one keyword, 'cutegingercat'. User is not allowed to change the name of this file or move it to another directory.

Extra restriction is applied in case user wants to download a post from Instagram that contains two specific hashtags (both hashtags present in the post). The user then should only put those two hashtags in `classes.txt` file. For example, `classes.txt` should have this structure:

```
hashtag1
hashtag2
```

4 Choosing Web to Download Contents From Number of Contents to be Downloaded

After running `main.py`, user will be prompted this on terminal:

```
(1) Google
(2) Instagram
(3) Twitter
Enter (1), (2), or (3):
```

Simply enter number 1 or 2 in the terminal (without the parentheses) to indicate the desired web.
After that, the user will be asked to enter the number of samples he/she wants to download from the chosen web.

```
Enter number of samples to be downloaded:
```

Samples can be images, videos, or both. The entered number indicates how many samples per keyword/hashtag that will be downloaded. It is important to note that depending on the keywords, the actual numbers of downloaded samples may be less than the entered number. When downloading samples from Instagram, it may also be the case that the downloaded images or videos are more than the entered number. The reason behind this behaviour, is because Instagram scraper download posts. A post in Instagram can contain multiple images and videos.

For downloading samples that contain two specific hashtags (only available for Instagram), the entered number will be the number of samples that contain both hashtags. For example, a user wants to download 30 posts that has both cat and dog hashtags, then 30 should be entered instead of 15.

5 Google Image Scraper

Google Image scraper only downloads images from Google Image. After the user chose Google and entered the number of samples to be downloaded, the program will take some time to download the images.

The program will create a path `/database/google` and inside the google folder, there will be folder(s) named according to the keyword(s). User can find the images of each keywords in the the corresponding folder. The images are named in `keyword_index` pattern, where keyword is the keyword for the image, and index is a unique number indicating the image ranging from 1 up to `n`, with `n` being the number of samples to be downloaded.

Additionally, user can find a descriptor file in each of keyword folders. A descriptor file describes the following properties of the downloaded images: caption, width, height. The line number of each descriptor corresponds to index of downloaded image.

Below is an example of how the directories look like when a user downloads 3 images related to cat:

```
database
--google
----cat
------cat_1.jpg
------cat_2.jpg
------cat_2.jpg
------google_cat_descriptor.csv
----google_log
--instagram
--twitter
```

6 Instagram Scraper

Beside downloading images, Instagram scraper has an extra feature, which is downloading videos. After the user chose Instagram and entered number of samples to be downloaded, the user will be asked to indicate if he/she wants to download posts in certain time period.

```
Download posts in certain time period?
(1) Yes
(2) No
Enter (1) or (2):
```

If a user wants to download posts within certain time period, the user will be asked to provide the range of date that is desired. For example:

```
Since year: 2019
Since month: 12
Since day: 12
Until year: 2019
Until month: 12
Until day: 31
```

Followed by a prompt asking if user wants to download posts based on hashtag(s) or based on username(s).

```
(1) Download posts based on hashtag(s)
(2) Download posts from user(s)
Enter (1) or (2):
```

After that, the user will be asked if he/she wants download only images, only videos, or both images and videos.

```
(1) Image only
(2) Video only
(3) Both Image and video
Enter (1), (2), or (3):
```

Followed by another prompt asking if user wants to download with one hashtag or two hashtag.

```
(1) Download with one hashtag
(2) Download with two hashtag
Enter (1) or (2):
```

User should only enter 2 if there are exactly two hashtags written in the `classes.txt` file.

The images and/or videos are named in `username_uploaddate` pattern.The `txt` caption files have the same naming pattern as the samples file name. Therefore it is easy for user to see the corresponding caption of a post. Example of a `txt` file named `lunali_1_2020-12-14_22-40-15.txt` is given below. `lunali_1` is the account that uploaded the post, while `2020-12-14_22-40-15` is the time that post was uploaded.

```
Mila the milanese
*
*
#dogs #dog #floof #brindle
```

Below is an example of how the directories look like when a user downloads one post with 'cat' hashtag and one post from a username called 'kyliejenner':

```
database
--google
--instagram
----hashtags
------cat
--------captions
----------_loversgifts__2021-01-06_20-44-32.txt
--------_loversgifts__2021-01-06_20-44-32_1.jpg
----users
------kyliejenner
--------captions
----------kyliejenner_2020-12-14_02-50-53_1.txt
--------kyliejenner_2020-12-14_02-50-53_1.jpg
--twitter
```

7 Twitter Scraper

Twitter scraper mainly scrapes tweets and download them into separate txt files. Beside that, if a tweet contains images, they will be saved as well. After the user chose Twitter and entered the number of samples to be downloaded, the user will be asked to indicate if he/she wants to download posts in certain time period.

```
Download posts in certain time period?
(1) Yes
(2) No
Enter (1) or (2):
```

If a user wants to download posts within certain time period, the user will be asked to provide the range of date that is desired. For example:

```
Since year: 2019
Since month: 12
Since day: 12
Until year: 2019
Until month: 12
Until day: 31
```

After that, the user will be asked if he/she wants to download tweets based on keywords or based on usernames:

```
(1) Download tweets based on keyword(s)
(2) Download tweets from user(s)
Enter (1) or (2):
```

The `txt` files that contain tweets are named in `tweetid_todaydate` pattern. A `txt` file contains the tweet, number of current likes, and the date that tweet was posted. The images that are downloaded are named in `tweetid_index` pattern, where index indicate the index of the image in case the tweet contains more than one image. An example of `txt` file name `1334663899572883457_2021-01-11.txt` is given below. `1334663899572883457` is the tweet id, while `2021-01-11` is the date that I downloaded this tweet.

```
A note from Billie on the \WHERE DO WE GO?" World Tour.
https://t.co/y23giu5agi
Likes: 67517
Posted at: 2020-12-04 01:00:45
```

Below is an example of how the directories look like when a user downloads one post with 'cat' keyword and one post from a username called 'billieilish':

```
database
--google
--instagram
--twitter
----keywords
------cat
--------media
--------1346923051829633028_2021-01-06.txt
----users
------billieeilish
--------media
----------1334663899572883457_0.jpg
--------1334663899572883457_2021-01-11.txt
```

Not all tweets contain an image, therefore it can be seen that `database/twitter/keywords/cat/media` is empty.