Instagram-crawler-python

Allowed library:

- BeautifulSoup
- requests
- Selenium
- Tkinter
- pandas
- threading

scenario:

1. Search for a favorite hashtag on Instagram and list the n accounts that contained these hashtags in Instagram search results. n must be parametric and configurable at the beginning of the program.

2. In the next step, consider the last m of each of these accounts and extract the text of the comments below each of these posts along with the number of likes. m must be parametrically adjustable.

3. Save the stored data in a data frame and finally on the hard drive. Your data-frame should contain a comment, the username of the author of the comment, the username of the main post account, and the number of likes of the comment.

4. A graphical user interface (preferably created with tkinter) that has the tools to interact with the user and display the desired output. Including:

    - In this window, the user should be able to see the list of accounts that your program will go to their posts and increase or decrease them.
    - Adjust the values of m and n.
    - This window should also show the user how many accounts have been crawled so far and how many are left.
    - The elapsed time and the estimated time to the end of the work should also be displayed.
    - At the end of the work, show a report of the time spent for the whole work.
    - The location of the output file on the hard drive is determined by the user.
    - Adding other items in this graphical interface will give you a score.

5. Your program should be written in multi-thread. You can choose to use a multi-thread to crawl each account or create a new thread to crawl each comment. If your program runs faster than your other classmates, it will score.

6. This program does not consider object-oriented programming style; But if this program is implemented objectively, it will have a score.

7. The functions and classes you define must have dock strings. The cleanliness of the codes is mandatory according to the principles introduced in the classroom.

Special points section:

Consider a specific issue and search for related hashtags through your app. For example, suppose you want to see how positive or negative people's comments about Samsung are. Search all the hashtags related to Samsung, including the relevant brands, etc. through your program. Extract the relevant comments.

Then consider 1000 random comments and tag them. In this way, if the comment had a positive opinion of Samsung, the label would be positive, if it had a negative opinion, it would have a negative label, and if not, it would have a neutral label. (Usually tagging up to 1000 comments takes less than 2 hours. You can do this part of the work with the help of your other classmates. The more comments in this section, the more accurate your output will be.)

Using the fasttext module in Python, you can create a model that is tagged from these 1000 comments to partially learn the pattern of positive and negative comments. This model can then express its opinion about the positive and negative of each new comment with the understanding that it has found a negative or positive opinion! So with this model you can test all the comments and check how many of the comments were positive or negative.

Your app can report how many user reviews have been positive or negative about this topic. (Or display as a percentage)

You can see the necessary explanations for doing this part of the project on this page.

Documentation:

You will also need to prepare a report file about your project, its features and its various sections. Upload the report file, along with project codes, extracted data, and the results of a sample execution, as a zip file.

Using Git has a score.