AUTOMATIC DATA MINING ON FACEBOOK

Midterm Report

SAJAD WAZIN

June 30th 2022

CONTENTS

LIST OF FIGURES

ABSTRACT

Garnering over 2.93 billion active monthly users [1], Meta's popular social media platform, Facebook, has become one of the easiest way to reach a public audience. This report will discuss the development of the application "FacebookWebScraper" in an effort to motivate independent researches on Facebook's suggestions algorithm and its participation in siloing* users.

* *siloing: isolate (one system, process, department, etc.) from others.*

1 DOCUMENTATION & REPOS

The project is based in a private GitHub repository that is set to follow a local branch where I develop code for the program. Being the sole developer for this project, the repo is mainly used as a means of publishing code for progress reports and project submissions. The repo also provides access to documentation pertaining to the type of information that can be scraped with the program and how it can be used for research.

2 SOFTWARE ARCHITECTURE

2.1 Programming Language

The programming language that was used for this project is Java. One of the main reasons behind this architectural decision is because of the expansion quality that Java offers. Being an Object-Oriented Programming language, Java allows developers to easily create data structures and interfaces that inherit from other code-bases.

SELENIUM The main library used in this project was Selenium, a web application automation library. A lot of the work done for FacebookWebScraper involves manipulating data from Selenium, which, at its core, is scraped as raw HTML elements. As a result, Java's OOP features are essential to writing code that inherit from Selenium, but also to modify and process the data that has been found.

SEPARATION OF CONCERNS In order to provide a software architecture that is usable, clean and can be expanded upon, the design principle of "separation of concerns" is unavoidable. In fact, this design principle was applied to Facebook-WebScraper with two main goals: (1) avoid inappropriate intimacy between functional elements and (2) avoid cyclic dependencies between functional elements. Java makes it extremely easy to assign proper access and design well-encapsulated objects to keep data safe, secure and organized. This can be seen through not only the class division of the project, but even its package division. By separating concerns adequately, FacebookWebScraper has been designed in such a way that the "gui" package is so well-encapsulated that it can actually be completely removed from the project without removing any functionality from the application.

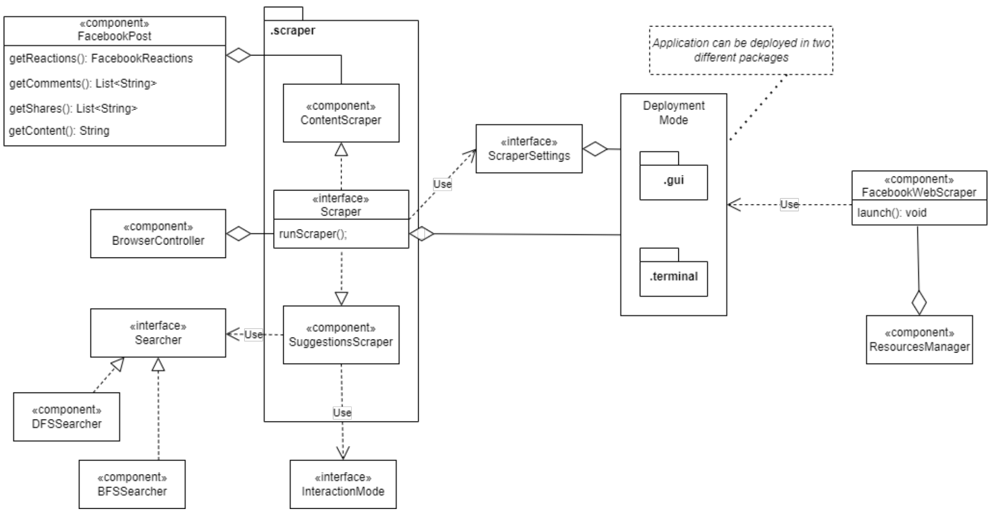

«component» FacebookPost
getReactions(): FacebookReactions
getComments(): List<String>
getShares(): List<String>
getContent(): String
.scraper
«component» ContentScraper
«interface» Scraper
runScraper();
«component» BrowserController
«interface» Searcher
Use
«component» SuggestionsScraper
«component» DFSSearcher
«component» BFSSearcher
Use
«interface» InteractionMode
Use
«interface» ScraperSettings
Application can be deployed in two different packages
Deployment Mode
.gui
.terminal
Use
«component» FacebookWebScraper
launch(): void
«component» ResourcesManager

Figure 1: A UML Functional Structure Model containing the important functional components of FacebookWebScraper

2.2 Libraries

The choice of libraries in this project is the compromise between the size of the application's deployment package and the amount of well-documented and tested external code that can be used. Thus, the libraries used in the project are only the essentials and were used as a foundation to be built upon.

2.2.1 *Selenium*

Selenium is the main driver in the development of the application. It contains all the code that allows to start up and control a web browser. FacebookWebScraper uses it to find "WebElements" that contain the relevant data. It mainly does so by using the page's XPath syntax to locate specific CSS properties in conjunction with a mix of JavaScript and Selenium's built-in web-interaction features to navigate different WebElements. The main features used from Selenium are the following:

1. Locating WebElements by ID and XPath
2. Clicking on WebElements
3. Running JavaScript commands and passing WebElements as arguments
4. Implicit waiting (sleep)
5. Explicit waiting (waiting until a WebElement is found)
6. Handling different tabs
7. Initializing the browser with custom settings

2.2.2 *JavaFX*

JavaFX is a framework for creating GUI applications in Java. Using JavaFX, the default way to launch and use FacebookWebScraper is through the GUI. It provides an interface for users to program different scrapes on Facebook without any coding background. Similarly, it also allows to sanitize the scraping settings by restricting what the user can input.

2.2.3 *JSON*

JSON provides an easy-to-use and readable file format to save the data gathered by the application. Using JSON and its Java library, FacebookWebScraper can encode post scrapes with the following information:

- The post URL
- The content of the post
- The reactions on the post
- The comments on the post
- The amount of shares on the post
- The captions of each share

2.3 Maven

Maven is an essential tool for the development of this application. It is a build automation tool that, in this case, is used for dependency management and artifact generation. It is used to fetch the required versions of Selenium, JavaFX and JSON for the project, without distributing the libraries as part of the code-base. Similarly, by using the Maven Compiler Build plugin, developers can run tests by simply compiling the project and running it through their IDE of choice*. In particular, this compiler plugin allows to easily set distinct Java versions for the source code of the project and the compiled target artifact.

Another important aspect of Maven is the Maven Shade plugin that is used to generate an executable ".jar" artifact for FacebookWebScraper. This Shade plugin allows to create "fat jars", which are Java artifacts that are compiled and packaged with all relevant dependencies and libraries. In this case, since Java no longer packages JavaFX for SDK versions after 11 and because Selenium and JSON are not part of the Java SDK, these dependencies must be built into the artifact to run properly. Similarly, since JavaFX requires inheritance from its "Application" class for it to launch, Maven can be used to direct the compiler to the main "Application" class.

* The GitHub repository contains a project-specific ".idea" folder respective to IntelliJ, but any IDE can be used to work on the project.

3 REVISION

3.1 First Progress Report

There have been a lot of changes since the first progress report and the current status of the project. For some aspects of the project, progress has gone much better than initially planned, while for other features, progress was much slower than expected, nigh impossible to complete.

3.1.1 GUI

The first milestone in the project was the design and conception of the GUI. In general, the development of the user-interface went almost perfectly according to plan, with no divergences from the original expected result. The only notable difference would be the expected completion date for the GUI; in this case, it was partially completed much sooner than expected, particularly by June 4th 2022. It is said to be "partially" completed because, as the expected features of the program keep changing, the GUI itself must be adjusted accordingly. However, the main functionality provided by the GUI, that of configuring scraping settings, is not expected to change and is near its final stage of development. This feature also leaves the "terminal" version of the application out of the project's scope, although accommodations will be made in the code to allow for future implementations of the feature.

3.1.2 Groups

As a result of personal testing, Facebook's suggestions algorithm for groups is very limited in the number of immediate outputs. Upon following or joining a group, the user is usually met with no suggestions. In cases where suggestions are provided, it still seems to be very limited, usually providing only one or two suggestions. On the other hand, the groups' "Discovery" tab seems to be populated over time, however, rarely updated immediately and not updated at all if the user fails to join or follow groups over a period of time. For this reason, performing a study on Facebook's groups suggestion algorithm is deemed unwarranted as the algorithm itself bears a limited number of outputs. Correspondingly, aiming part of FacebookWebScraper's development towards groups is unmerited because no conclusive findings can be made with the proposed features in the first progress report. In conclusion, the program's development no longer takes into account any type of "search" or "scraping" when it comes to Facebook groups.

3.1.3 Post Content Scraping

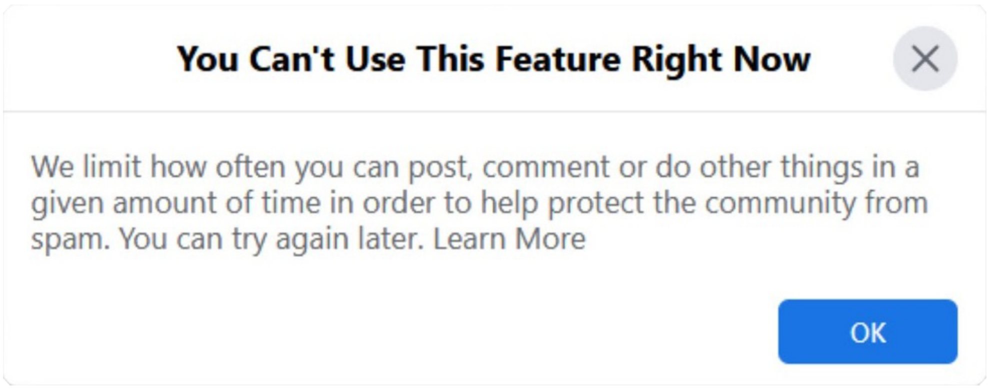
The initial plan for scraping content on Facebook would have been to create two different interfaces: one for scraping content on Facebook Pages and one for Facebook Groups. This has now changed to being a single interface, that for scraping posts on any Facebook link containing a feed of posts. The reason for this change is a result of the lack of groups suggestions [3.1.2], therefore not justifying a personalized interface for scraping groups. However, since the interface only requires access to a feed of posts, the door to groups scraping is still open for future developers of FacebookWebScraper. This interface was completed much later than planned, with the milestone being reached on June 21st 2022.

3.1.4 Encoding

The initial plan for the program failed to mention any notion of "encoding" data or persistence as it was deemed trivial. This is an aspect of the software that turned out to be much trickier than anticipated, eventually taking days as opposed to what was planned to be a few hours. The main issue was, and still is, encoding and saving data as it is parsed through the scraper. This issue is a trade-off between efficiency and program security. If every bit of data gets saved upon discovery, then the program becomes slow due to waiting for I/O operations. If information is only saved at the end of a scrape, then the program risks losing all data in the events of a crash. As of now, the selected solution is the latter option for simplicity. During testing on over 600 posts, the program has not encountered a crash a single time, thus ruling out the immediate need for asynchronous I/O operations. As mentioned in section 2.2.3, the encoding solution uses a JSON format to save all of the post's information. This was deemed to be the most optimal strategy because JSON is widely supported across most popular programming languages, and because it is also one of the easiest file formats to encode and decode. Currently, the encoder only supports data gathered through post scraping as it is the only feature that is fully complete. The current non-implemented design for encoding data gathered through suggestions scraping will follow a tree-like structure with "parent" nodes, being the page that has been followed, pointing to its "child" nodes, being the page suggestions.

3.1.5 *Suggestions Scraping*

The suggestions scraping algorithm, being the main purpose for the development of this application, has also become the main block in development. As reported in section 3.1.2, searching through groups suggestions is now unfeasible. For pages, the BFS searching algorithm has been written*, and through testing, displays no flaws or crash possibilities. However, the issue for suggestions scraping is Facebook's Bot prevention algorithm. Albeit harmless, FacebookWebScraper is rightfully identified as a bot by Facebook after following a certain number of pages. Although Facebook sometimes displays a prompt allowing the user to continue despite the accusation, it is very likely that being flagged multiple times would lead in the account's deactivation.


You Can't Use This Feature Right Now

We limit how often you can post, comment or do other things in a given amount of time in order to help protect the community from spam. You can try again later. Learn More

Figure 2: An example of Facebook's bot prevention warning prompts

3.1.6 *Research and Data Analysis*

Doing research and data analysis is no longer part of the project. The reason for this is due to the lack of time to be allocated for research. As of writing the midterm report, the time taken for program development has already surpassed 150 hours, much more than what was initially planned for this project. In section 3.2.2, it will be explained that another participant of the "AI & Society" research group may take over in this section of the project.

* In theory, the algorithm works, however, due to bot prevention, it has not been tested thoroughly.

3.2 Current Plan

This section will cover the current direction of the project, the end goal and how everything will be completed. This current plan is a result of the findings about Facebook's limits [2] and the time allocated for completing this project.

3.2.1 *Project Description*

The new direction this project will be taking is mainly focused on software development. FacebookWebScraper is expected to be delivered, complete with a working GUI, a post scraping algorithm and a DFS and BFS searching algorithm for Facebook Pages. The development of the application is heavily fixed on software architecture, to allow for the software to be used by technology enthusiasts and researchers. Similarly, this attention to software architecture also allows future programmers to work on this project and develop new and better features for FacebookWebScraper. As mentioned in sections 3.1.1, 3.1.3 and 3.1.4, the code-base already welcomes many new features to be implemented. The official end goal of the project is now to provide a highly customizable, easy-to-use and expandable application that allows anybody to scrape information on Facebook.

3.2.2 *Division of Labour*

This project will be complete once the software has been fully developed with the remaining pieces constituting the Encoding system [3.1.4] and the Suggestions Scraper [3.1.5], both of which will be completed by me, Sajad Wazin. However, Tianyu Han from the "AI & Society" research group has shown interest in making use of this application for independent research once the project is complete.

3.2.3 *Deliverable Schedule*

The deliverable schedule will be flexible and heavily dependent on Facebook's bot prevention system. As of now, the application is fully functional and no new features will be added. What remains to be done in the development process is the following:

1. Communicate with Tianyu about features to add for researching purposes and implement them if needed (July 1st)
2. Write scripts to conduct research with the data that can be gathered (TBD)
3. Find a workaround for decreasing the number of botting flags that the application will raise during Suggestions Scraping on Facebook (Until project ends)
4. Design and implement an encoding system for data gathered during Suggestions Scraping (July 5th)
5. Fault-proof the application to cover all possible crashes during scraping and design an error-handling system to keep data safe in such cases (July 10th)
6. Persistence Service to save the encoded files and load/sanitize input files (July 12th)
7. Update the GUI according to the new plan. (Until project ends)

4 DEMONSTRATION

The demonstration that will be presented for the midterm report will be a rudimentary "features" test. This demonstration will display the following features:

1. A working GUI allowing the users to program scrapes
2. A post scraper that can save the content, comments, reactions, and shares of any given post.
3. An encoding system for the result of scraping posts.
4. A partially complete BFS searching algorithm for page suggestions.

REFERENCES

[1] Meta Platforms, Inc. Meta Earnings Presentation Q1 2022, 2022.

[2] Elfsight. Facebook Limits (groups, friends, invites etc) and Account Blocks, 2017.