Bot Crawler to retrieve data from Facebook based on the Selection of Posts and the Extraction of User Profiles

Bot Crawler para la obtención de datos de la red social Facebook a partir de la Selección de Publicaciones y Extracción de Perfiles de Usuarios

DOI: http://doi.org/10.17981/ingecuc.18.2.2022.08

Artículo de Investigación Científica. Fecha de Recepción: 13/09/2022. Fecha de Aceptación: 20/09/2022.

Ariel Guillermo Sánchez Paipilla
Universidad Pedagógica y Tecnológica de Colombia. Sogamoso (Colombia)
ariel.sanchez@uptc.edu.co

Mónica Katherine Durán Vaca
Universidad Pedagógica y Tecnológica de Colombia. Sogamoso (Colombia)
monica.duran@uptc.edu.co

Javier Antonio Ballesteros Ricaurte
Universidad Pedagógica y Tecnológica de Colombia. Tunja (Colombia)
cjavier.ballesteros@uptc.edu.co

Ángela María González Amarillo
Universidad Nacional Abierta y a Distancia. Tunja (Colombia)
angela.gonzalez@unad.edu.co

Pedro Nel López Castellanos
Universidad Pedagógica y Tecnológica de Colombia. Sogamoso (Colombia)
pedronel.lopez@uptc.edu.co

To cite this paper:
A. Sánchez Paipilla, M. Durán Vaca, J. Ballesteros Ricaurte, A. González Amarillo & P. López Castellanos, "Bot crawler to retrieve data from Facebook based on the selection of posts and the extraction of user profiles", *INGE CUC*, vol. 18, no. 2, pp. 101–113s. DOI: http://doi.org/10.17981/ingecuc.18.2.2022.08

Resumen

Introducción— Los datos se pueden encontrar dentro y fuera de las organizaciones; y crecen exponencialmente. Hoy en día, la información disponible en internet y las redes sociales se ha convertido en un generador de valor a través del análisis efectivo de una situación específica y el uso de técnicas y metodologías que permiten proponer soluciones basadas en contenido para así poder implementar procesos de toma de decisiones oportunos, inteligentes y asertivos.

Objetivo— El objetivo principal de este trabajo es el desarrollo de un rastreador web que permita la extracción de información de Facebook sin restricciones de acceso o el requerimiento de credenciales, el cual estaría basado en rastreo web y técnicas de raspado a través de la selección de etiquetas HTML para identificar y definir patrones.

Metodología— El enfoque utilizado para el desarrollo de la presente propuesta implicó 4 etapas principales: A) Trabajo colaborativo SCRUM; B) Comparación de técnicas de extracción de datos en la web; C) Extracción y validación de permisos para el acceso a los datos en la red social Facebook; y D) Desarrollo del Bot Crawler.

Resultados— Como resultado de este proceso, se creó una interfaz gráfica que permite revisar el proceso de obtención de datos derivados de perfiles de usuario en esta red social.

Conclusiones— Para la obtención de datos de la red social Facebook a partir de la selección de publicaciones y extracción de perfiles de usuarios, el tiempo de ejecución del Bot Crawler se optimiza de manera considerable respecto a otras APIs, donde a mayor obtención de perfiles que acceden a una publicación semilla, menor tiempo de obtención de datos.

Palabras clave— Raspado web; rastreo web; HTML; redes sociales; datos

Abstract

Introduction— Data can currently be found within organizations and outside of them, they are growing exponentially. Today, the information available on the Internet and social networks has become a generator of value, through the effective analysis of a specific situation, using techniques and methodologies with which content-based solutions can be proposed, and thus achieve, execute timely, intelligent and assertive decision-making processes.

Objective— The main objective of this work is to development of a Bot Crawler, which allows extracting information from Facebook without access restrictions, or request for credentials, based on web crawling and scraping techniques, through the selection of HTML tags, to track and be able to define patterns.

Methodology— The development of this project consisted of four main stages: A) Teamwork with SCRUM, B) Comparison of web data extraction techniques, C) Extraction and validation of permissions to access the data in Facebook, D) Development of the bor crawler.

Results— As a result of this process, a graphical interface was created to review the process of obtaining data derived from user profiles in this social network.

Conclusions— As a result of this process, a graphical interface is created that allows checking the process of obtaining data derived from user profiles of this social network.

Keywords— Web scraping; web crawling; HTML; Social Networking; data

I. Introduction

The power of the data is increasing in the technological revolution. The data on the web, the social networks and the cloud with specific characteristics allow proposing solutions based on content that generates value, differentiation and improvement opportunities [1]. This is possible with the use of algorithms, techniques and methodologies that allow for understanding quantitative and qualitative structured, semi-structured, and unstructured data in the form of web pages, HTML tables, web databases, emails, tweets, blog posts, photos, videos, among others, for their subsequent retrieval or analysis [1]-[2].

Nevertheless, with the increasing growth of the information available on the internet, it becomes increasingly difficult to find strategies to obtain and manage the data generated on a daily basis. Thus, algorithms are used to crawl information that access web servers' resources without human intervention [3].

The algorithms search the data on the internet, analyze the content, and save information on indexes and databases [4]-[5]. They allow to collect data automatically and continuously, such as contact details and profiles, web pages viewed, and links clicked, among others, with diverse purposes such as marketing or data mining for personal, social or business decision-making processes.

Two algorithms used for crawling information are the web crawler and the web scraper, which are processes or bots for the automatic exploration of web pages [1]. The first one organizes the content into indexes and assesses it by accessing the URLs, analyzing the content and indexing the new links. The second one extracts the specific data from the websites and stores them in databases for their posterior analysis [6].

These algorithms require constant adjustments due to the dynamics of the information, the existence of various structures in the web pages [7]-[8], and the limitations in the access to the information of certain websites, such as some social networks. These characteristics make unified extraction difficult, and periodically scanning the content and keeping it catalogued for use is an increasing challenge.

This is why it is necessary to develop an algorithm that allows one to obtain or extract information from Facebook with unrestricted access to the public data of user profiles without requiring credentials to download information and allowing the selection of tags with the selected content.

This paper is structured as follows: Section 2 presents the analysis of the current situation, Section 3 presents the state of the art where previous works similar to this research are discussed, Section 4 presents the methodology where the phases of the project are described, Section 5 presents the results of each of the processes completed in the project, and Section 6 presents the conclusions.

II. Analysis of the Current Situation

Data have become an essential part of research because of the possibility of obtaining an added value when analyzing them [1]. The web offers all types of information in different formats, such as blogs, websites and social networks. However, it is impossible to see all the data simultaneously on one website since they are distributed on several pages in different sections.

Most websites do not allow storing a local copy of the data. Therefore, the only option is to manually copy and paste the data shown by the website, which makes the process longer and more complex.

In the case of social networks such as Twitter, Reddit and Instagram, specific data are accessed through the Application Programming Interface (API) by sending a request to the server [9]. However, since Facebook is the biggest social network in the world, it has the API Graph, which has more limitations. For example, it cannot be used without having a created app, and applications cannot be created due to server failures. Additionally, it has restrictions to access the data [10], due to the misuse of the data in the Cambridge Analytica scandal [11]. This misuse made them take action to protect the data of the user profiles, making it difficult to collect and access data in a conventional way.

To automate the process of obtaining data and achieve an easier and more effective way to analyze or even visualize the information, techniques such as web scraping are used, through which data can be extracted from multiple websites to a single database. It is used to convert unstructured data from the web into a local centralized database [3]. Through APIs, data collection from these websites is fast and can be done without any web scraping software.

Another technique associated with web scraping is web crawling, which collects information from the URLs, meta tags, web page title, etc., covering large sets of web pages. Its goal is to search and extract web pages and web documents that are specific and only relevant to a given topic or keyword according to the user's requirements instead of downloading all web pages like a traditional search engine [12].

Considering the limitations in obtaining data from Facebook when using the automated extraction techniques mentioned [13], it is necessary to develop a bot crawler that allows obtaining data from the profiles of this network that react to a post without access restrictions or credentials to download data, only by accessing with the user and password of the personal profile.

III. State of the Art

The following works directly relate to this research based on the development of automated crawlers. Moreover, they evidence the necessity of designing a bot crawler that allows crawling data in social networks without restriction or limitation in the access to the data.

HRBUST (China) presents a study of the analysis and data crawling based on social networks through the design of web scraping programs to crawl information and comments in China's biggest social network, Sina Weibo, similar to Facebook [12]. This study concludes that access to the API is restricted for data collection, and the amount of collected data is minimum.

Cu and TUAS (Turkey) present an extensive analysis of the available social networks and their APIs and describe the design of a social network crawler as support in the Natural Language Processing (NLP) area [14]. It includes conclusions on the limitations and restrictions of the generic crawlers that allow access only to public content.

BTH (Sweden) and UCDavis (USA) developed a crawler called SINCE, which has characteristics superior to other crawlers [15]. Regarding efficiency and crawl depth, SINCE interacts with each unique post allowing support to make informed decisions on what content to re-crawl.

RPS Group of Institutions (India) [16] present the construction of a "spider" that crawls and indexes Facebook using crawler programs that proved to be an effective tool for data extraction.

PUJ (Colombia) developed an app to support cyber intelligence management, which allows one to search and collect information on the deep web, dark web and some pre-selected social networks [17]. This app enhances the information security strategy and the decision-making processes in companies and organizations.

After reviewing these works, it can be concluded that it is important to use data extraction techniques, but it is also necessary to develop a proprietary automatic crawler that allows access and extraction of all the data from the profiles that react to a post made on a social network.

IV. Methodology

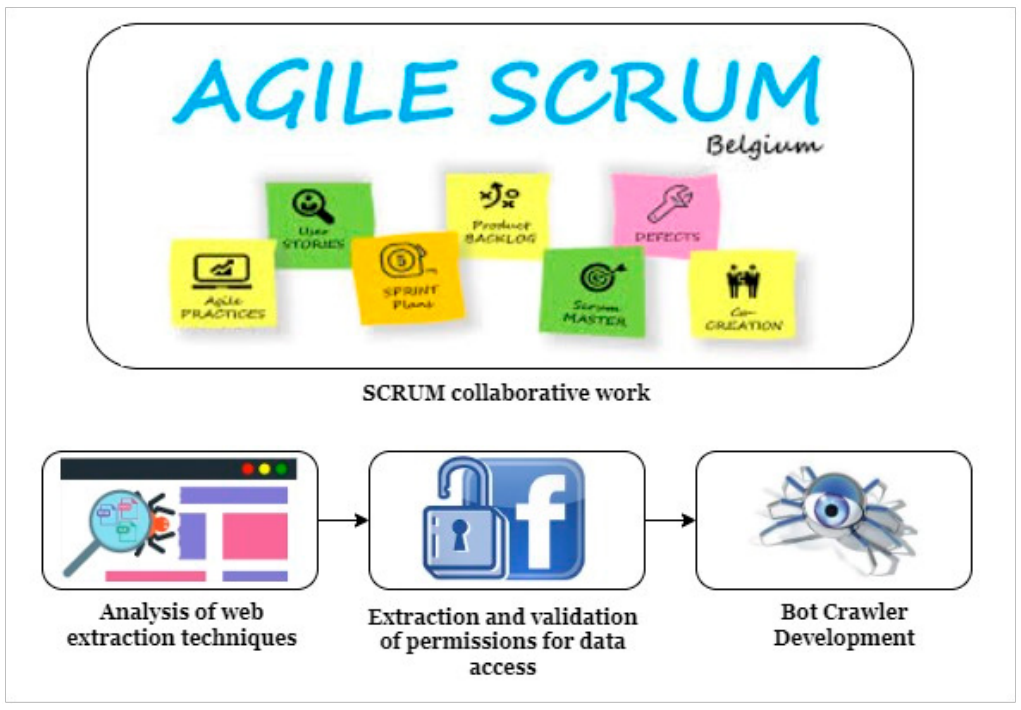

Fig. 1. Methodology.
Source: Authors.

The development of this project consisted of four main stages: A) Teamwork with SCRUM, B) Comparison of web data extraction techniques, C) Extraction and validation of permissions to access the data in Facebook, D) Development of the bot crawler. The methodology is shown in Fig. 1 and explained next.

A. *Teamwork with SCRUM*

This stage, transversal to the other stages, is based on the agile methodology SCRUM, which, with the collaborative work approach [18], addresses a set of good practices to obtain better results. The central axis for fast deliveries in fixed times encompassing the entire work process is based on short execution cycles or iterations called "sprints" [19]. In each sprint, there is a deliverable or enhancement of the product that provides value to the customer. These sprints usually last between two and twenty-four weeks.

This methodology proposes three roles [19]: the Product Owner, who is in charge of maintaining the vision of the product and is clear about what the development team is building; the Scrum Master, in charge of advising and providing the necessary information to the work team; and the Development Team in charge of making the product, they have to be able to self-organize and choose how to develop the work.

Based on these work roles in the development of the project, fifteen (15) sprints were carried out, distributed as follows: two (2) sprints for the comparison of web data extraction techniques, five (5) sprints for the collection and validation of permissions to obtain data from Facebook, six (6) sprints for the development of the algorithm with the validation of the functionalities and the development of the web application, and the remaining two (2) sprints were aimed at the refinement and testing of the final algorithm.

B: *Comparison of web data extraction techniques*

Two techniques are used for extracting and obtaining information from websites, which are sometimes mistaken because their methods are related. However, their objective is different: web scraping and web crawling [20]. These techniques automatically simulate the navigation of a human on the web through HTTP requests to the server and applications where the options for tracking web pages are customizable. They capture information from web pages and record the identified links so that they can be used to locate new pages and obtain the data contained in these sites [21].

The web scraping technique is the extraction process used to automatically collect relevant data from websites, converting unstructured information into structured information for further analysis. As CAETI and UAI (Argentina) says [20], it is related to web indexing, which indexes information using a robot, and it is a universal technique adopted by most search engines. However, web scraping focuses more on transforming unstructured data on the web than on structured data that can be stored and analyzed in a central database.

Some restrictions when using this technique are: the security in each of the web pages, blocking of requests by each server when detecting that it is accessed from a domain external to those hosted in the server, and it is not feasible to access any data of a web page being a limitation for obtaining the data. Therefore it is not useful to some extent because some pages block when you try to make an HTTP request, which means that even if the web scraping is implemented correctly, only the metadata will be accessible.

Web scraping is responsible for obtaining the HTML tags and, based on libraries such as BeautifulSoup and using filters such as find or findAll to search in one or several tags, stores these sections in an array that divides them for subsequent review and verification of each fraction by boxes to access the content.

Another technique used for information extraction is the web crawling technique that allows only the raw retrieval of an HTML, a raw file type. As stated by LPU, ACR (India) and MDX (UK) [22], it is a process performed to collect web pages to index them and use them to display search results according to user requirements. This technique results in a small repository of words in an accessible storage.

The main restriction when using any of the mentioned techniques is that they do not allow to obtain complete information since they do not select the total of HTML tags.

C. *Extraction and validation of permissions to access the data on Facebook*

Access to public data in social networks such as Facebook is very limited. The current extraction techniques are not enough to extract the data due to the security since it is not possible to access without verified credentials, and the APIs only allow access to certain data. There is the option of requesting from the personal profile a "Facebook developer" role, a request that can take between 3 to 6 months approximately, and that, when authorized, allows access to the data only of the groups or companies created by that profile, this being a disadvantage when obtaining data from any profile or group.

Another drawback of accessing Facebook data is that when scrolling through the pages looking for information, the system blocks the HTTP requests that are responsible for making the request to obtain an HTML response. They respond to headers that are the Link, CSS and Scripts tags, shown in Table 1, related to the general information of a page, which are useful for web positioning, accepted language of the page, etc. These headers have content that is not useful for possible analysis because it is only basic information such as titles, the author of the page, and web server response times.

Table 1.
Headers from Facebook.

Headers	Example
HTTP header User-Agent	Mozilla/5.0 (X11; Linux x86_64; rv:12.0) Gecko/20100101 Firefox/12.0
HTTP header Accept-Language	en-US
HTTP header Accept-Encoding	gzip, deflate
HTTP headers Accept	image/*
HTTP header Referer	http://www.google.com/

Source: Authors.

The headers used when web scraping a web page, e.g. *User Agent* or *Accept HTML*, are responsible for making the HTTP request to a page. The request could be made without these headers, but some websites request them to block bots and limit access to their information.

Thus, it is not enough just to add the headers in the request, since any server implements Cross-Origin Resource Sharing (CORS), which, as it says CISPA (Germany) and CNRS (France) [23], is a mechanism that allows restricted resources to be requested on a web page from a domain different from the domain that served the first resource. In other words, it is responsible for limiting which web domains (HTTP or HTTPS) can consume application server services, making it even more difficult to obtain data.

Through HTTP requests to web servers, data can be obtained when the server is not highly secure. This means there are no blocking requests from a console or command terminal since only a response is obtained from the <body> tag of the web page selected for data extraction. This tag contains the data necessary for analysis and storage.

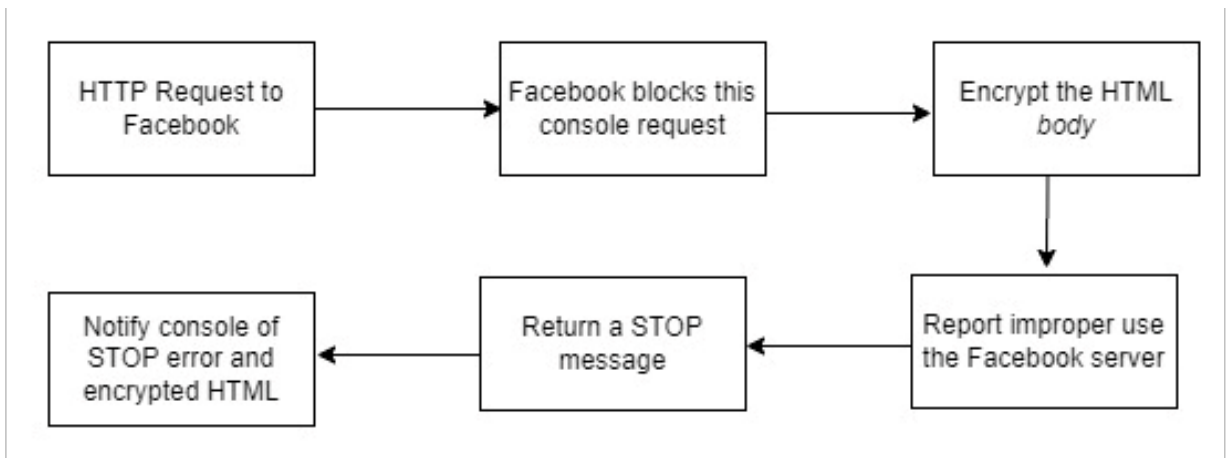

Fig. 2. Flow of the attempt to obtain data from Facebook with web scraping.
Source: Authors.

In the case of Facebook, as shown in Fig. 2, when trying to obtain data with this same HTTP request using the web scraping technique, there is usually a high level of security and access to an unauthorized resource is blocked. The server detects that this request is not a system call and blocks itself by returning the STOP message and an HTML code that encrypts the <body> tag restricting the extraction of its content. Only the content of the <head> tag can be retrieved.

The Content-Type property can be found when obtaining the data for a basic server, which contains the TEXT/HTML configuration in charge of converting HTML to text for manipulation. On the Facebook server, we add the charset="utf-8", which is in charge of special characters such as accents; the "Cache-Control" adding the private setting, which indicates that the response can only be stored by private caches; no-cache, which indicates that the response can be cached but must be validated with the origin server before each reuse; no-store allows a client to request that the caches refrain from storing the request; and the response and must-revalidate indicate that the response can be used as long as it is fresh but that once the resource becomes obsolete, it further restricts access to the data.

In addition, the Facebook server contains a report-to property responsible for reporting to an endpoint that is Facebook's own communication point, which is responsible for sending the request data to one of its systems about an improper request that was generated in the system.

The difference in security between the basic server and the Facebook server is evident because obtaining data from this social network is not so easy or straightforward since it cannot be accessed with traditional methods such as API's and consumption through HTTP requests with external libraries.

It should also be considered that when visiting a web page with an HTTP method, what is obtained is the initial interface without rendering any component, which limits the search for information using normal requests, which means that the bot can simulate the behavior of a person, not be detected and activate in real-time components to obtain this data.

The following libraries, available in Python, were used to validate the permissions for accessing the data on Facebook:

- *Selenium*: For unit test automation to simulate human behavior on the web. It generates a graphical client of the browser that is configured in this case with Google Chrome, which will be in charge of receiving requests and going through the pages automatically.
- *Time*: If many requests are made to a server at once, the server detects that a human cannot do this and immediately generates a temporary block to restrict access to the data. This python library is used to make time between requests to avoid detection.
- *Pyatogui*: For keyboard manipulation and commands to be executed for the operation of data inspection.
- *BeautifulSoup*: To extract data from HTML and XML files.
- *Request*: To make requests to the server and request page changes.
- *Mongo*: The pymongo connector is implemented for data storage in MongoDB.
- *Os*: For file directory recognition and loading multiple files into a data array for further processing.
- *Hquery*: Php library for the web requests to a server and obtaining the response in different data formats.

Knowing the way to access the data of Facebook and the validation of permissions through known information extraction techniques, it should be clarified that none of these techniques, web scraping or web crawling, are related to hacking methods considered as vulnerations to the security of pages looking for flaws in the system to exploit them. Using the extraction techniques mentioned above, only public data is obtained. The data are not accessed, or the access to the data is not violated since the data is on the web page and can be viewed by anyone.

D. *Development of the bot crawler*

For the development of the bot crawler, it is necessary to evade the previously mentioned security to avoid being detected. The bot is divided into three essential steps to obtain the data.

1) Enter a seed post (URL) to obtain the profiles that react this post: In this first step, there is a base parameter, the URL of a Facebook post, i.e. a seed post placed by the bot administrator containing an image as a base and profiles that react to that post. This URL is the bot's main route to access, start the search process, and obtain the data.

   The seed publication allows access to the algorithm developed for profile collection, executing it from the local terminal. The algorithm "notifies" the browser to open a new tab, where the bot indicates the Facebook login link to access and log in with user authentication elements for permissions and credentials. After each login through the test automator with Selenium, it is necessary to open a session again, considering that the access has already been made with the permissions of the profile, and thus, access to the seed publication is granted.

   Once the seed publication has been accessed, the browser must be "persuaded" to think that whoever is executing the actions or requests to the server is not a bot but a person who is browsing. The bot waits for some time to execute requests, showing that it cannot make a certain number of clicks in a second or more to access some page or link. This gives the server time to gather more data and get the profiles that reacted to that specific post. Fig. 3 shows the diagram for this step.

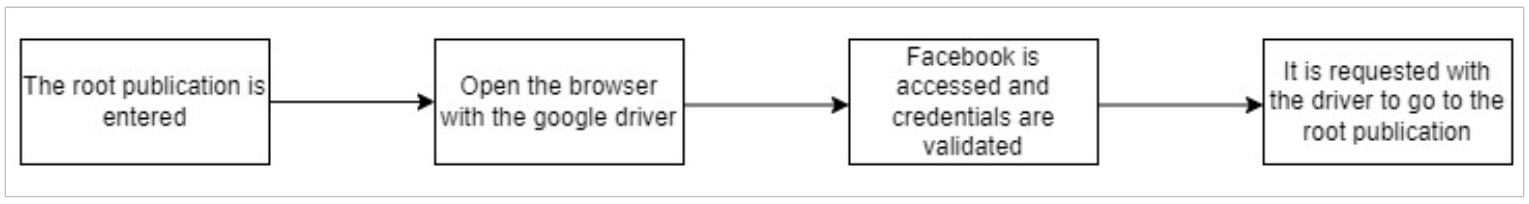

Fig. 3. Demonstration of the data collection.
Source: Authors.

To obtain the profiles that reacted to the seed publication, it is necessary to activate the pop-up windows of content of the profiles that reacted to the post. UseContext is used for this purpose, which is in charge of modifying the status of the pop-up by changing the property set{Name_PopUp} to the value TRUE to activate it and thus show it to the user. Once these pop-ups are activated, the first ten profiles are obtained by default since, due to the high data load, it only shows paged information in each request.

An important aspect of executing the developed algorithm is that the browser screen is limited to that of a cell phone since web scraping requires the largest number of tags to track the information more easily. Due to the responsive designs of web pages, a web design technique that seeks the correct display of a page on different devices, new tags are placed to assign new styles, allowing more tags to track in a better way the desired elements. In this case, the browser screen is set to a resolution of 400 × 700 pixels.

Each time the algorithm is executed, a counter is incremented by 1 when a request is made (through the pyautogui library) to obtain data. Fig. 4 shows how the counter is updated 4 times only when the posts obtained are less than those requested in the previous request. In this way, the algorithm knows when there are no more data and stops to avoid consuming more resources.

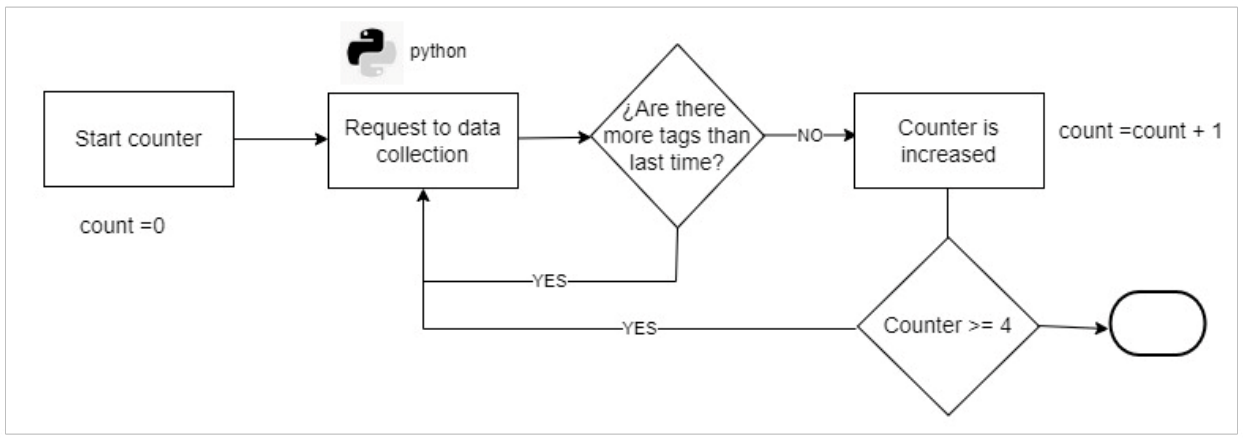

Fig. 4. Diagram of the start of the counter through the pyautogui library.
Source: Authors.

2) Analysis of profiles obtained by going through the profile of each person who reacted to the seed publication and download in HTML format: in this step, the bot crawler is in charge of reading the list of profiles that reacted to the seed post from step 1. It goes through each profile and requests more information than the one loaded by Facebook, which allows making more requests in each profile to obtain more data.

   After the request for each profile, the algorithm executes keyboard commands to go to the bottom of the page, repeating this 10 times by default and thus forcing the browser to load more information than it allows by default. After completing these 10 requests, the algorithm has a time-out of 30 seconds in each profile to download and store the information of each of them in an HTML file (Fig. 5). Additionally, this time-out is implemented to make the browser think a person is accessing the data, not a bot. This is how the profiles for the next step are obtained (3).

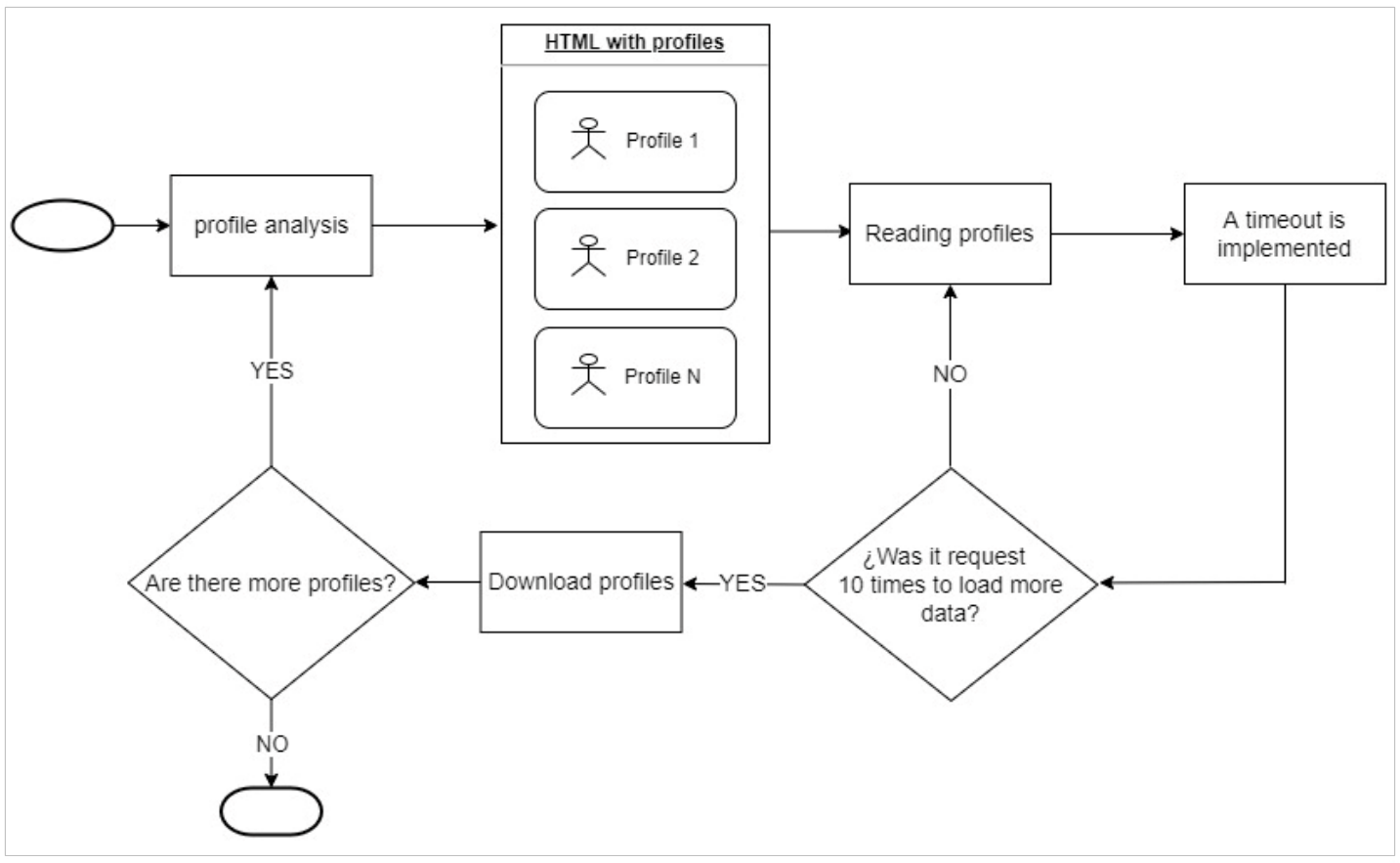

Fig. 5. Download of the obtained profiles.
Source: Authors.

3) The data of each person is obtained, and the web scraping filters are applied to store the data in MongoDB: in Step 3, the bot crawler is in charge of analyzing the downloaded tags in HTML format of each profile to do the web scraping. The name of each profile and the written text are extracted. Then, a JSON file is created to store this data in the MongoDB database. Fig. 6 shows a diagram explaining the selection of tags for each profile, building the JSON, and storing the information in MongoDB.

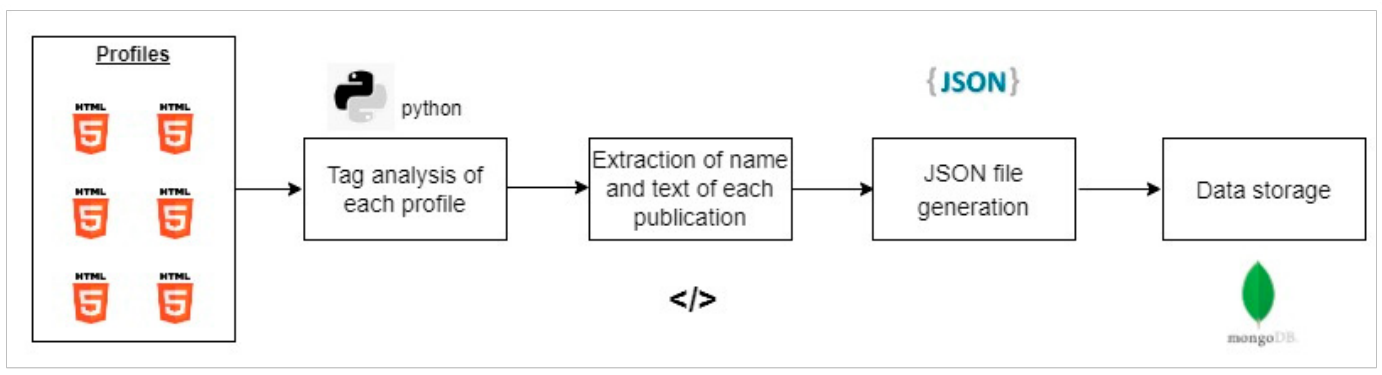

Fig. 6. Selecting the tags for each profile, building the JSON, and storing the information in MongoDB.
Source: Authors.

V. Results

As a result of the bot crawler execution, 15 seed posts were selected, to which approximately 700 profiles reacted. Subsequently, with the implementation of the web scraping, the HTML tags of each profile were analyzed, and around 13 030 entries were extracted and stored in the MongoDB database. Fig. 7 shows the process to this result.

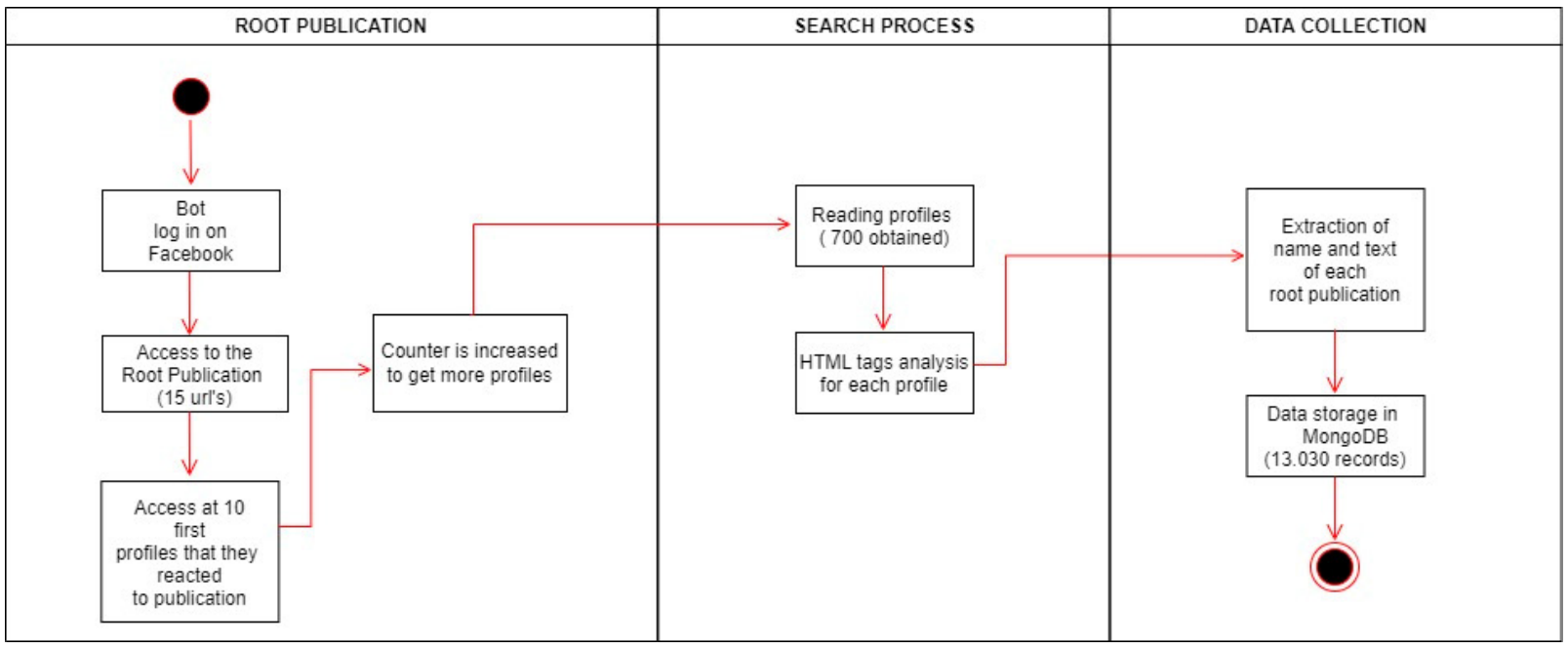

Fig. 7. Diagram of the entries selection result.
Source: Authors.

The data obtained from each profile were stored in a semi-structured way without a predefined schema in a JSON format file from data collections, where each of them works as an autonomous information unit. Three attributes were downloaded from the profiles: *Profile ID*, which refers to the identification of the user profile on Facebook; *Profile Name*, which is anonymized for information security; and *Content*, which stores the comments made to the seed publication. Fig. 8 shows an example of some collections.

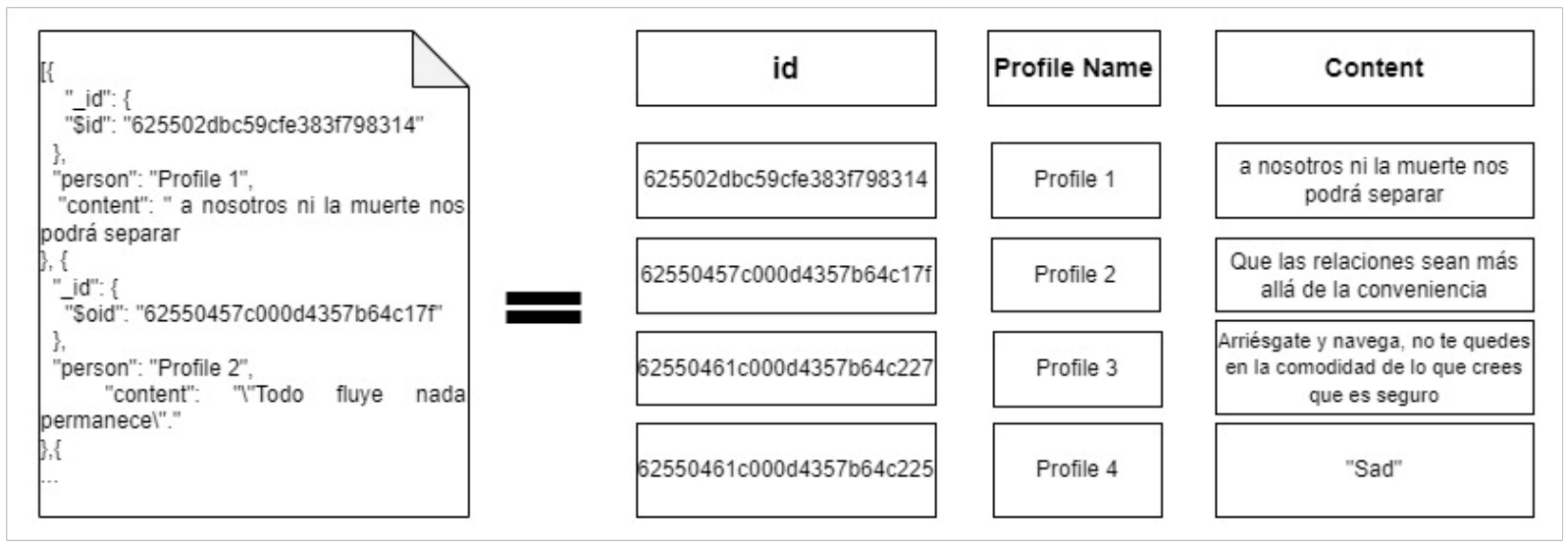

Fig. 8. JSON file example.
Source: Authors.

The seed publications selected from public Facebook groups deal with suicidal behavior and ideation. This topic was selected since this study is part of the development of the second activity of the methodology of the research project, "Análisis del comportamiento suicida en redes sociales mediante analítica de datos y machine learning" [Analysis of suicidal behavior in social networks through data analytics and machine learning]. Fig. 9 shows a word cloud visualization in which the most significant words can be identified from the total number of stored records, where the size is larger for those that appear more frequently.

Fig. 9. Word cloud.
Source: Authors.

To test the functionalities of the bot crawler, the information obtained from Facebook is validated by means of a web application since the bot is only managed from the terminal, and it is difficult to read tags without formatting.

The web application has an approach to streamline the process of web scraping where it is possible to filter by the tags desired from an input and obtain all its contents, such as their classes, which allow to crawl and define patterns, being the latter the true objective of the analysis.

To start using the web application interface, it is necessary to enter the URL of a seed profile for the data collection validation. Subsequently, the tag to be filtered is entered, initially by a general tag, for example , which is the one that contains most of the tags, allowing a global selection of the entire seed profile.

If the objective is making a specific selection, for example about which images each profile contains, the <img> tag should be selected to obtain better results. In this example, the web application is in charge of rendering the obtained image and not only displaying the link, which allows a better understanding of the information when searching or filtering.

When executing the HTTP request service from the web page, three attributes are taken from the response JSON returned by the service. These are generated through each request to the seed profile for analysis, which are:

- *Position*: When searching in an HTML obtained through the bot crawler, it is required to know in which position is what the user is looking for since the tags are repeated several times.
- *HTML*: Each tag has classes for general styles or unique identifiers, and this allows to find patterns for the characteristics to be filtered during the web scraping, facilitating the work and reducing the time since it is necessary to track everything from the console.
- *View*: To display images, but the graphical interface is rendered according to the content to be displayed, either text or images.

From the above, it can be concluded that there is a need to find patterns in the labels that will lead to the generation of specialized filters for people with advanced knowledge. This allows access from a larger tag to get more content, such as having news cards where it is necessary to extract not only a name but complete content. In addition, the web application has a metadata module that allows obtaining data such as:

- *Description*: Contains a summary description of the information on the loaded page.
- *Keywords*: Contains words separated by commas, the keywords related to the content of the page.
- *Author*: Information on the author of the web page. The author can be seen as the administrative contact of the web.
- *Copyright*: Information on the copyrights of the web page.

Each web page contains specific tags to rank high in search engines. This is also called SEO (search engine optimization). The relevance of metadata for SEO is that search engines use it to obtain information about the web page.

The SEO is found in the <head> tags, a tag that Facebook does not encrypt, which makes it possible to obtain and extract figures on its performance. From the HTTP request to the Facebook server, an HTML is obtained, but the <body> tag is encrypted for security reasons.

Even so, tags such as <title> and others can be accessed which are shown in the metadata found within the <head> tag of each page as shown in Fig. 10.

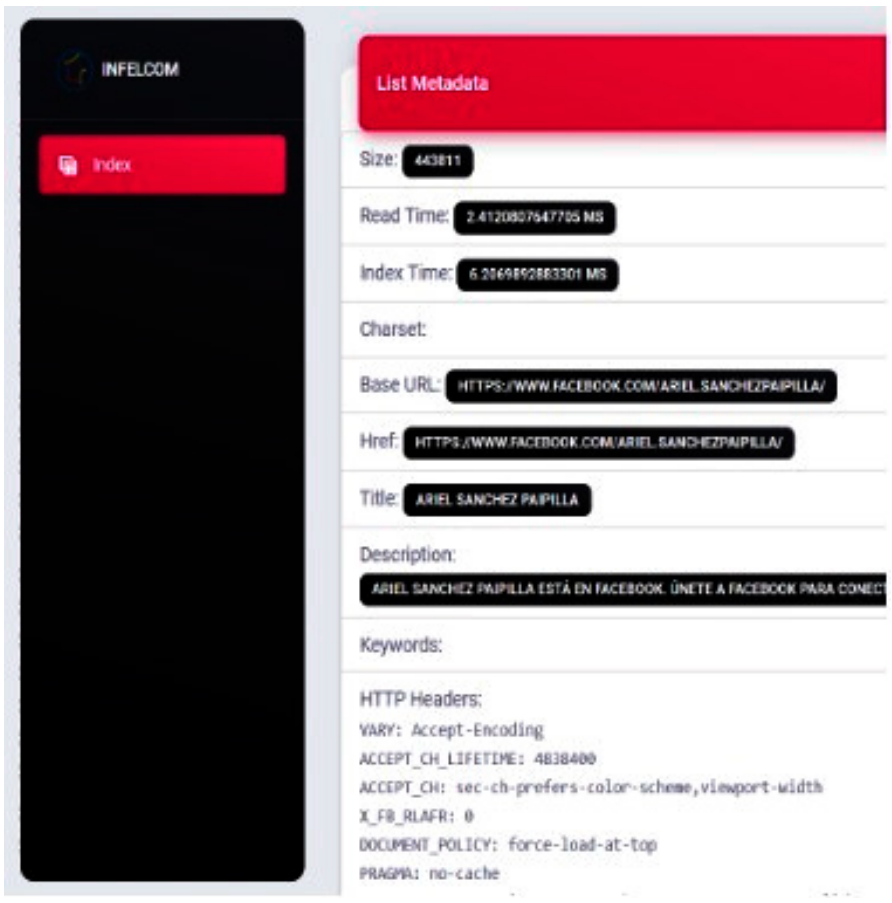

Fig. 10. Methodology.
Fuente: Autores.

VI. Conclusions

The development of this project allowed the creation of a bot crawler to extract and access public data of user profiles on Facebook without requiring credentials to download information and allowing the selection of tags that store the selected content. The results of the first version of the algorithm are promising with respect to data extraction with ethical rigor and scraping efficiency in web environments.

Unlike the Facebook API, the bot crawler allows the download of more information since it does not implement paging in the data; therefore, there is no limitation of access to them. In addition, a drawback of the API is that when querying too much data, it executes a block of up to one hour to retrieve it again, a drawback that is not present in the bot crawler.

To obtain data from Facebook from the selection of posts and extraction of user profiles, the execution time of the bot crawler is considerably optimized with respect to other APIs, where the more profiles that access a seed post, the less time it takes to obtain data.

Future work can explore using the records obtained by the bot crawler to perform a text analysis in which Natural Language Processing (NLP) techniques are applied, allowing the information to be processed to establish relationships, classifications or predictions.

Financing

Scientific research article derived from the research project "Analysis of suicidal behavior in social networks through data analytics and machine learning", financed by the Pedagogical and Technological University of Colombia. Start year: 2021, end year: 2022.

Acknowledgements

The authors thank the psychologists of the National Open and Distance University and the Pedagogical and Technological University of Colombia, for the explanations and clarifications on the key aspects to understand suicidal ideation.

References

[1] N. Bolbol & T. Barhoom, "Mitigating Web Scrapers using Markup Randomization," presented at *2021 Palestinian International Conference on Information and Communication Technology*, PICICT, GZA, PS, 28-29 Sept. 2021. https://doi.org/10.1109/PICICT53635.2021.00038

[2] L. Wang & H. Wang, "Design and Research ofWeb Crawler Based on Distributed Architecture," presented at *3rd International Conference on Artificial and Advance Manufacture*, AIAM, MAN, UK, 23-25 Oct. 2021. https://doi.org/10.1145/3495018.3495061

[3] P. Thota & E. Ramez, "Web Scraping of COVID-19 News Stories to Create Datasets for Sentiment and Emotion Analysis," presented at *14th Pervasive Technologies Related to Assistive Environments Conference*, PETRA, CFU, GR, 29 Jun. 2 Jul. 2021. https://doi.org/10.1145/3453892.3461333

[4] H. Habib, S. Pearman, E. Young, I. Saxena, R. Zhang & L. Cranor, "Identifying User Needs for Advertising Controls on Facebook," presented at *Human-Computer Interaction*, ACM, NYC, NY, USA, 2022. https://doi.org/10.1145/3512906

[5] M. Klymash, I. Demydov, L. Uryvskyi & Y. Pyrih, "A Brief Survey on Architecture of Feedback Systems for Interactive E-Government ICT Platforms," presented at *15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering*, TCSET, LV-SLA, UA, 25-29 Feb. 020. https://doi.org/10.1109/TCSET49122.2020.235475

[6] A. Lagopoulos, G. Tsoumakas & G. Papadopoulos, "Web robot detection: A semantic approach," presented at *30th International Conference on Tools with Artificial Intelligence*, ICTAI, VLS, GR, 5-7 Nov. 2018. https://doi.org/10.1109/ICTAI.2018.00150

[7] M. Hossen, Y. Wang, H. Tariq, G. Nyame & R. Nuhoho, "Statistical analysis of extracted data from video site by using web crawler," presented at *2018 International Conference on Computing and Artificial Intelligence*, ICCAI, CHD, CN, 12-14 Mar. 2018. https://doi.org/10.1145/3194452.3194466

[8] Y. Feng, J. Li, L. Jiao & X. Wu, "BotFlowMon: Learning-based, Content-Agnostic Identification of Social Bot Traffic Flows," presented at *2019 IEEE Conference on Communications and Network Security*, CNS, WA D.C., WA, USA, 10-12 Jun. 2019. https://doi.org/10.1109/CNS.2019.8802706

[9] P. Lewandowski, M. Janiszewski & A. Felkner, "SpiderTrap - An Innovative Approach to Analyze Activity of Internet Bots on a Website," *IEEE Access*, vol. 8, pp. 141292–141309, Jul. 2020. https://doi.org/10.1109/ACCESS.2020.3012969

[10] J. Ho, "Assessing the bias of Facebook's graph API," presented at *30th ACM Conference on Hypertext and Social Media*, ACM, HOF, DE, 17-20 Sept. 2019. https://doi.org/10.1145/3342220.3344923

[11] Y. Huang, "Privacy Security Status and Countermeasures in the Era of Big Data," presented at *3rd International Conference on Big Data Engineering and Technology*, BDET, SGP, SGP, 16-18 Jan. 2021. https://doi.org/10.1145/3474944.3474952

[12] G. Gao, Y. Liu & G. Bai, "Crawling and Analysis of Data Based on Social Networking on Stock Comments," presented at *IOP Conference Series: Earth and Environmental Science*, IOP, HB, CN, 14-16 Dec. 2019. https://doi.org/10.1088/1755-1315/234/1/012093

[13] Ö. Çoban, A. Inan & S. Özel, "Facebook Tells Me Your Gender: An Exploratory Study of Gender Prediction for Turkish Facebook Users," *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 20, no. 4, pp. 1–38, Jul. 2021. https://doi.org/10.1145/3448253

[14] S. Pais, J. Cordeiro, R. Martins & M. Albardeiro, "Socialnetcrawler - Online social network crawler," presented at *11th International Conference on Management of Digital EcoSystems*, MEDES, LMS, CYP, 12-14 Nov. 019. https://doi.org/10.1145/3297662.3365805

[15] F. Erlandsson, R. Nia, M. Boldt, H. Johnson & S. Wu, "Crawling Online Social Networks," presented at *2015 Second European Network Intelligence Conference*, ENIC, KKN, SE, 21-22 Sept. 2015. https://doi.org/10.1109/ENIC.2015.10

[16] M. Yadav, G. Tanwar & A. Wadhwa, "Social Network with Web Crawler & Cluster," *Int J Comput Sci Commun*, vol. 10, no. 2, pp. 171–179, Mar. 2019. Available from http://csjournals.com/IJCSC/PDF10-2/4.%20Meenu.pdf

[17] G. Colmenares, N. Méndez y O. Virgüez, "Deep Dark Web & Social Crawler (DDW&SC): Aplicativo para apoyar la gestión de Ciberinteligencia", *Trabajo de grado*, Fac Ing, Prog Ing Sist, PUJ, BOG D.C., CO, 2019. Available: http://hdl.handle.net/10554/47278

[18] M. Ramírez, M. Salgado, H. Ramírez, E. Manrique, N. Osuna y R. Rosales, "Metodología SCRUM y desarrollo de Repositorio Digital," *RISTI*, no. E17, pp. 1062–1072, Ene. 2019. Disponible en http://www.risti.xyz/issues/ristie17.pdf

[19] B. Grebić & A. Stojanović, "Application of the Scrum Framework on Projects in IT Sector," *Eur Proj Manag J*, vol. 11, no. 2, pp. 37–46, Dec. 2021. https://doi.org/10.18485/epmj.2021.11.2.4

[20] R. Martínez, R. Rodríguez, P. Vera y C. Parkinson, "Análisis de técnicas de raspado de datos en la web aplicado al Portal del Estado Nacional Argentino", presentado al *XXV Congreso Argentino de Ciencias de la Computación*, RedUNCI, Rio CTO, AR, 14-18 Oct. 2019. Disponible en http://sedici.unlp.edu.ar/handle/10915/91026

[21] I. Galdino, E. Gallindo & M. Moreira, "Utilização de Bots para Obtenção Automática de Dados Públicos usando as Técnicas de Web Crawling e Web Scraping," presentado a *VIII Workshop de Computação Aplicada em Governo Eletrônico*, WCGE, POA, BR, 16-20 Nov. 2020. https://doi.org/10.5753/wcge.2020.11269

[22] S. Kaur, A. Singh, G. Geetha & X. Cheng, "IHWC: intelligent hidden web crawler for harvesting data in urban domains," *Complex Intell Syst*, pp. 1–19, Jul. 2021. https://doi.org/10.1007/s40747-021-00471-1

[23] G. Meiser, P. Laperdrix & B. Stock, "Careful Who You Trust: Studying the Pitfalls of Cross-Origin Communication," presented at *ACM Asia Conference on Computer and Communications Security*, ASIA CCS, HK, CN, 7-11 Jun. 2021. https://doi.org/10.1145/3433210.3437510

Ariel Guillermo Sánchez Paipilla. Systems and computing engineer at the Pedagogical and Technological University of Colombia. His work focuses on data analytics, computer science and telecommunications in the INFELCOM-UPTC research group. His favorite hobby is software research and development. https://orcid.org/0000-0001-7181-1466

Mónica Ketherine Durán Vaca. Systems Engineer from the Juan de Castellanos University Foundation. Specialization in Databases from the Pedagogical and Technological University of Colombia. Master in Information Engineering from the Universidad de los Andes (Colombia). She currently teaches at the Pedagogical and Technological University of Colombia in the area of Databases. INFELCOM-UPTC research group. She is interested in analytics and data science topics. https://orcid.org/0000-0002-4806-683X

Javier Antonio Ballesteros-Ricaurte. Systems Engineer from the University of Boyacá (Colombia). Master in Computer Sciences agreement between the Autonomous University of Bucaramanga (Colombia). Technological and Higher Studies Institute of Monterrey (Mexico). Professor at the UPTC School of Systems and Computing Engineering and director of the Information Management Research Group. https://orcid.org/0000-0001-9164-4597

Ángela María González-Amarillo. Systems Engineer Juan de Castellanos University Foundation (Colombia). Specialist in Higher Distance Education National Open and Distance University (Colombia). Master of Business Administration National Open and Distance University Florida (USA). Professor of the systems engineering program at the Universidad Nacional Abierta y a Distancia Tunja (Colombia). Zonal leader of the School of Basic Sciences, Technology and Engineering of the Boyacá Center Zone of the National Open and Distance University, GIDESTEC Research Group. https://orcid.org/0000-0002-1825-0097

Pedro Nel López Castellanos. Systems Engineer from the University of Boyacá (Colombia). Specialization in Design and Construction of Telematic Solutions from the Autonomous University Foundation of Colombia. Specialization in Financial Management from the Jorge Tadeo Lozano University (Colombia). Master's Degree in Management Strategic Telecommunications UNINI (Puerto Rico). Professor at the Pedagogical and Technological University of Colombia, Research Group INFELCOM - UPTC. https://orcid.org/0000-0002-2219-7794