General Algorithm for Searching User Data in Social Media of the Internet

Andii M. Peleshchysyn, and Oleh V. Mastykash

Abstract—The research work presented within this paper solves the problem of automated search for heterogeneous data in social media of the Internet (SMI). Building a system for obtaining and subsequent analysis of heterogeneous data in SMI—a complex multi-stage process in which specialists of various profiles and qualifications participate. Therefore, one of the main problems in the design of such systems is the coverage of all aspects of the functioning of the software-analytical complex, providing a common language for specialists, which allows us to uniquely, and clearly, understandably formulate the basic concepts of the projects. One of the main and basic tasks in analyzing the pages of a SMI user is to build algorithms for analyzing the user data environment (UDE). The quality of software will depend on the implemented algorithms. The construction of such algorithms, on the one hand, provides an understanding of the process of forming functional individual modules of the system and their interaction, on the other hand, laying a qualitative foundation in the future system. Algorithms for data analysis in the SMI will be designed based on the basic principles of behavior of the user registered in it.

Index Terms—Data Search, Social Environment of the Internet, Database, Data Validation, User Data Environment.

I. INTRODUCTION

Today, various techniques and approaches for the analysis of SMI are presented. The most common analysis approach is an approach that depicts a SMI as a graph of interconnected nodes [1], whose vertices are users and the edges reflect the relationships between them. This approach is the most popular, since graph theory has provided a dictionary of concepts for the mathematical description of the properties of SMI. Finding social connections on the Internet is today a fairly well-solved task [2, 3]. Its solution can be carried out with varying degrees of automation, from full automation [4] to manual collection of information.

As described in article [5], the social environment of the Internet consists of user pages that are interconnected. The necessary conditions for constructing the data search algorithm of such users are the following conditions:

1) The availability of constructed data search models [6].
2) The presence of a general algorithm for constructing a hierarchical tree of SMI pages.
3) The presence of a built-in algorithm for parsing the user’s SMI page.
4) The presence of algorithms for saving and structuring the received data.
5) Availability of a designed database.

II. FORMATION OF A SET OF INPUT DATA FROM THE USER

To implement the conditions, it is necessary to obtain from the user a piece of information in which it will be indicated what exactly should be found and analysed [7]. Therefore, the initial stage of the algorithm will be the stage of generating input data for the search. Input data contains two types of signs: main and auxiliary. The main signs are signs by which the user can be found. Auxiliary signs are signs that are not used on their own, but supplement the main. The input data set will be formed in accordance with the constructed article [8]. The model of the input data set is shown in Fig. 1. The user can enter additional parameters manually, while specifying the parameters that must be used to search for these signs. If in the future the parameter is successfully identified and found, it will be added to the input data set.

The general search algorithm for heterogeneous data in social media of the Internet is shown in Fig 2.

The necessary criteria for the search, the user can select and fill in manually or select from the template, with its subsequent filling. Information will be entered through a UI application (Android, Windows). Available search options:

1) Highly specialized search. Specifies the specific community through which the search will be carried out.
2) Widely specialized search. Search all available communities.
3) An approximate search. When previous searches have not yielded results, it will be possible to select this type of search.

In all cases, the user has the flexibility to customize the criteria (attributes) by which the search will be performed.
After the data has been entered, it is necessary to check it for correctness. Data verification provides a better implementation of the following steps. Data verification will take place in several stages (Fig. 3):

1) Syntax analysis. Definitions between words of grammatical relationships. This stage will be implemented by a separate processing module, which filters options that do not constitute a semantic load and are illogical. Each feature entered will be checked separately, for example e-mail has been entered the phone number will be checked by regular expression.

2) Semantic analysis. Determination of the desired meaning of each word. At the same time, in the knowledge base, each word is attached to it with a specific meaning depending on the meanings of the surrounding words. To implement this stage, an external library will be used (move the paragraph below). Will not apply to all features.

After the input data set has been generated, you can proceed to the stage of selecting sources (SMI), which will be used to search for data. Information search sources will be added by administrators.

III. THE ALGORITHM FOR OBTAINING AND ANALYZING DATA IN SOCIAL MEDIA OF THE INTERNET.

The general data search algorithm is shown in Fig. 2 and consists of the following steps:

1) Select a template and enter information for the search.
2) Validation of data.
3) A breakdown of the input data into atomic sets in accordance with the selected search criteria.
4) Obtaining pre-stored data and checking its relevance.
5) The selection of SMI for which search and analysis will be carried out;
6) The selection of key nodes of the pages of the SMI that will be software-generated.
7) Downloading information.
8) Structuring, validating and filtering the received data.
9) Saving data.
10) Return of the analyzed data to the end user.

An information entry template is an entity that contains pre-selected nodes (criteria) of an input data set with partial or full content to search for SMI actors. The template can be created by the administrator and any registered member of the program. The template created by the administrator will be accessible to all participants. The template created by the user will be accessible only to him.

The algorithm for adding a new source is shown in Fig. 4 and consists of the following steps:
1) Enter the domain name of the SMI.
2) Enter user data on behalf of which program will run.
3) Enter the required headers for the queries.
4) Introduction of additional information to study the program (features of the SMI).

The authorization request contains the following data:
1) URL of the page.
2) Headings.
3) Cookies.
4) Parameters that are transmitted as part of the navigation line or in the request body.
5) Additional parameters (ajax, data of the house tree).

After user authorization on the community server, a session is opened for him. The response from the server includes the following data:
1) Parameters. Server variables on the basis of which the DOM tree of the page is formed.
2) Cookies.
3) Status response code from the server.

Typically, the session identifier is stored in cookies or in a tree-house (depending on the principles of the community). A user token can be transmitted either in cookies or as one of the answer options. The token [10], session identifier and other necessary parameters (browser version, etc.) will be checked by the server for each user request to identify the user. If the request fails, the server will return an error code.

The token is a digital signature of the user's login and password. In different technologies, the token generation is different in algorithm and principle, for example: bearer token, access_token, token, custom_token and so on. But they are united by a common look: a pair of <key, value>.

An example of a token is shown in Fig 5.

The token lifetime is determined on the server side of the community. Most often, it is deleted when a user session is deleted. The token is one of the main keys to access the SMI data, but not the only one. In addition to this parameter, other header keys are also used. When adding a new resource, the administrator must enter them manually.

**TABLE I: APPLICATION AUTHORIZATION REQUEST**

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters</td>
<td>username, password, headers (ajax, data of the house tree).</td>
</tr>
<tr>
<td>Location</td>
<td><a href="https://www.facebook.com/">https://www.facebook.com/</a></td>
</tr>
<tr>
<td>Set-cookie</td>
<td>secure:httponly; tko=6ooHQQmoUYy76e,AWX5rG_uqKXuVuAfrkEd60C; expires=Wed, 17-Jan-2018 12:22:32 GMT; Max-Age=777600;</td>
</tr>
<tr>
<td>Content-type</td>
<td>text/html; charset=UTF-8</td>
</tr>
<tr>
<td>Vary</td>
<td>Origin</td>
</tr>
</tbody>
</table>

The algorithm for adding a new source is shown in Fig. 4 and consists of the following steps:
1) Enter the domain name of the SMI.
2) Enter user data on behalf of which program will run.
3) Enter the required headers for the queries.
4) Introduction of additional information to study the program (features of the SMI).

Since it was described in [9], almost all SMI hide their information from unregistered users. To access community data, the program must have a set of appropriate rights. Therefore, for the program to work correctly, it is required that it access the platform on behalf of a registered user. To successfully receive the source document of a user’s page, the program must first go through authorization. The authorization procedure for the user looks like this:
1) Go to the login page.
2) Enter the username and password.
3) Fill in other necessary data in the appropriate fields (captcha, secret key ...).

After successful authorization, the user is redirected to a personal page in the community. The procedure for authorizing an application in the community is slightly different:
1) Obtaining the url of the login page.
2) Formation of a request with a username, password and other necessary data.
3) Sending a request (POST request) to this page and waiting for a response from the server.

If the answer contains information about successful authorization, then it is necessary to save the token [9] and other necessary data for further work of the program. An example request sent to the community to authorize the application is shown in Table I.
After entering this data, the SMI server checks the correctness of the input by sending a request to the target resource, if the answer is correct (for example, the status of the response code will be 200) [11], then this resource is added to the lists of available sources. The program can also automatically generate headers based on the received response / answers.

IV. THE ALGORITHM FOR SEARCHING DATA ON THE PAGES OF THE SMI

The first step of this stage is the formation of the page structure by which we will search. If the search function for the user according to the specified criteria is present in the SMI, then we form a request to the search page, which in the body will contain data intended to refine the search and send a request to the community to search for the user according to the entered data.

We save links to pages of found profiles. If there is no such search functionality, then we use a recursive search for the user's page throughout the community. Starting point we can take a user on behalf of whom we are contacting the community (the algorithm is described below). We also launch a separate module in parallel, which will look for references to the user and the entire community (the algorithm will be described below). The algorithm for the formation of the structure of the pages from which we will receive the information is shown in Fig.6.

If the person is in the database, then read the information it and show it to the user. At the same time, we indicate the date when these data were received. If the user specified the updated data, then we send requests to all personal pages of the SMI that are stored in the database and re-read the data. If for some reason the user can no longer be found at this address, then we proceed to search for it in the SMI.

Search for a new user in the community:
1) We go to the search page for people in the community (its address must be stored in the database). We fill in the necessary parameters for the search, send a request, we get a response.
2) If the user was found, then save the path to his page in the database.
3) If no results are found, then we begin to look for mentions of a person in the entire community and save the addresses of resources (messages, comments, photos).
4) The answer can be three options: the person’s page is found or the page is not found, but there are references to the person and information about the person is not found.

V. DESIGNING A DATABASE FOR STORING INFORMATION

Relationship design plays an important role in any social network designed to connect people [12].

The most common database designs for social networking sites include quick reference functions, features for the addition and removal of social media sites and channels to your own database and the ability to filter sites based on multiple features.

The base scheme that allows you to save system information is shown in Fig 7.

If the person is in the database, then read the information it and show it to the user. At the same time, we indicate the date when these data were received. If the user specified the updated data, then we send requests to all personal pages of the SMI that are stored in the database and re-read the data. If for some reason the user can no longer be found at this address, then we proceed to search for it in the SMI.

Search for a new user in the community:
1) We go to the search page for people in the community (its address must be stored in the database). We fill in the necessary parameters for the search, send a request, we get a response.
2) If the user was found, then save the path to his page in the database.
3) If no results are found, then we begin to look for mentions of a person in the entire community and save the addresses of resources (messages, comments, photos).
4) The answer can be three options: the person’s page is found or the page is not found, but there are references to the person and information about the person is not found.

The database scheme that allows you to save user information is shown in Fig 8.

It is inefficient to store media files in a database. Such files can be stored on a separate file server. In the database we will keep only links to these files. To optimize disk space, duplicate files will not be saved. But there is a high probability that after all we will save the same files. Therefore, in order to avoid this problem, a separate service will work in the background, which deletes the same files
and merges the links to them in the database.

Fig. 8. Designing a database. Part 2

Good database design is essential to building scalable, high-performance applications. A database is nothing more than a mass of information stored in a framework that makes it easier to search. Everything else is just details. If a database works well, bits of related information are filed automatically and details can be pulled out needed. It should be simple to draw new meaning from data by compiling it into reports and visualizations, then storing away those facts for later use. Within that simple definition, there is infinite variation. Small decisions in the beginning have a huge cumulative impact.

VI. CONCLUSION

So the article implemented algorithms for searching user data in social media of the Internet. The main tools for constructing algorithms were used a unified modelling language UML. The construction of the algorithms was carried out taking into account the following aspects of system design: the design of the software-algorithmic complex was carried out from the point of view of the user, the input data was divided by their characteristics, the system was divided into separate independent modules, as well as decomposition of works.

The following algorithms are constructed in the article: The general algorithm for obtaining and analyzing data in the SMI, the algorithm for receiving and analyzing data in social media of the Internet and the algorithm for searching data on the pages of the SMI. And also a database was designed that will allow you to save the received information from the SMI.

REFERENCES


A. M. Peleshchynshyn was born on May 21, 1973, in a family of scientists. His father is Peleshchynshyn Mykola Andriyovych, and brother is Rostyslav M. Peleshchynshyn. He is Ukrainian scientist, doctor of technical sciences, professor, head of the Department of Social Communications and Information Activities of the Lviv Polytechnic National University, head of the scientific and methodological subcommittee on specialty 029 “Information, library and archival affairs” of the higher education sector of the Scientific and Methodological Council of the Ministry of Education and Science of Ukraine.

Since 1995 – worked as an intern researcher of the department of information systems and networks, Lviv Polytechnic National University, 1996-1999 – studied at postgraduate school stationary department of information systems and networks. From 1999 to 2001 – worked as an assistant. Since 2001 – worked as assistant professor of the same department. During 2008-2011 – worked as a professor of the department of information systems and networks. Since 2011 – is the head of the department of social communication and information activities of the Lviv Polytechnic National University. Since 2016, he is the head of the scientific and methodological subcommittee on specialty 029 “Information, library and archival affairs” of the higher education sector of the Scientific and Methodological Council of the Ministry of Education and Science of Ukraine, he is the developer of educational standards in the specialty. In November 2016 he was awarded a diploma of the Lviv Polytechnic National University for significant achievements in educational and scientific work.

O. V. Mastyakash was born on September 14, 1988, in Lviv. He received the B.Sc. degree at the Lviv Polytechnic National University in 2010, the M.Sc. degree at the Lviv Polytechnic National University in 2011. He is a postgraduate at the Department of Social Communications and Information Activities of the Institute of Humanities and Social Sciences of the Lviv Polytechnic National University. He started his academic career as teacher (since 2010) on the Department of Social Communications and Information Activities of the Lviv Polytechnic National University.