Big Data Analytics and the Social Web: a Tutorial for the Social Scientist

Schatten, Markus; Ševa, Jurica; Okreša-Đurić, Bogdan

Veröffentlichungsversion / Published Version
Zeitschriftenartikel / journal article

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Big Data Analytics and the Social Web

A Tutorial for the Social Scientist

Markus Schatten, Jurica Ševa and **Bogdan Okreša-Đurić**
Artificial Intelligence Laboratory, Faculty of Organization and Informatics
University of Zagreb
Croatia

Date of submission: April 27th, 2015 *Date of acceptance: July 23rd, 2015*

Abstract

The social web or web 2.0 has become the biggest and most accessible repository of data about human (social) behavior in history. Due to a knowledge gap between big data analytics and established social science methodology, this enormous source of information, has yet to be exploited for new and interesting studies in various social and humanities related fields. To make one step towards closing this gap, we provide a detailed step-by-step tutorial on some of the most important web mining and analytics methods on a real-world study of Croatia's biggest political blogging site. The tutorial covers methods for data retrieval, data conversion, cleansing and organization, data analysis (natural language processing, social and conceptual network analysis) as well as data visualization and interpretation. All tools that have been implemented for the sake of this study, data sets through the various steps as well as resulting visualizations have been published on-line and are free to use. The tutorial is not meant to be a comprehensive overview and detailed description of all possible ways of analyzing data from the social web, but using the steps outlined herein one can certainly reproduce the results of the study or use the same or similar methodology for other datasets. Results of the study show that a special kind of conceptual network generated by natural language processing of articles on the blogging site, namely a conceptual network constructed by the rule that two concepts (keywords) are connected if they were extracted from the same article, seem to be the best predictor of the current political discourse in Croatia when compared to the other constructed conceptual networks. These results indicate that a comprehensive study has to be made to investigate this conceptual structure further with an accent on the dynamic processes that have led to the construction of the network.

Keywords: big data analytics, social web, web mining, social and conceptual network analysis, natural language processing, social science, Croatian political blogging site

Corresponding Author: Dr. Markus Schatten, Assistant Professor
Affiliation: Artificial Intelligence Laboratory, Faculty of Organization and Informatics
Address: Pavliska 2, 42000 Varaždin, Croatia
e-mail: markus.schatten@foi.hr

European Quarterly of Political Attitudes and Mentalities - EQPAM, Volume 4, No.3, July 2015, pp.30-81.
ISSN 2285 – 4916
ISSN–L 2285 – 4916

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

1. Introduction

Never has there been so much data about social behavior of people, and never has this data been as accessible as today. According to (Statista, 2014) approximately 1.8 billion Internet users are using social networking applications in 2014 with Facebook having more than 1.44 billion active users. An average Internet user spends an average of 1.72 hours on social applications on a daily basis (Mander, 2015). It is estimated that the Library of Congress stores up to 462 terabytes of data. According to a prediction by the Internet Data Corporation in 2015 there will be 7.9 zettabytes of data available, which is the equivalent of 18 million Libraries of Congress (Century Link, 2011). A big portion of these data comes from social media and social interaction of people on-line (Roe 2012, updated by authors):

- Twitter has 500 million tweets per day or approximately 115 MB/sec of data created (April 2015)
- Facebook has 1,44 billion users, with 65% logging in daily (March 2015)
- LinkedIn has over 347 million users (January 2015)
- Facebook collects an average of 600 terabytes of data every day or more than 200 petabytes per year, and has more than 300 petabytes in one cluster (April 2014)
- 196.3 billion emails were sent daily in 2014
- There were 247.8 million blogs only on Tumblr and WordPress in 2014
- YouTube has 4 billion views per day (January 2012), 300 hours of video are uploaded per minute (April 2015)

The social sciences and the humanities have a natural interest in analyzing social web data where billions of people, on millions of sites, across thousands of diverse cultures, leave "trails" about various aspects of their lives (Schatten 2011). If we adopt the perspective of Niklas Luhman that social systems are systems of communication (Luhman 1984) it becomes obvious that by studying these "trails" (which we might interpret as the result of a social system's structural coupling to social media) we can analyze the social system itself: politics, culture, media, business, technology etc.

The current chapter is in a way a continuation of a series of articles published in the EQPAM journal (Neumann et al. 2014; Voinea & Schatten 2015) and especially (Schatten et al. 2015) in which we have provided a detailed introduction to social web mining and big data analytics methodology offering example studies for a number of important methods useful in the social sciences. After a very good reception of these papers we have decided to make one step further and provide a hands-on guide of a sample study on an actual social media site. In the process of conducting this study we have implemented a number of easy

ISSN 2285 – 4916
ISSN-L 2285 - 4916

to use tools which are available on-line for anyone to use. All tools, interim and final datasets, as well as results are listed in the appendix section.

For a sample study, we have decided to use the Croatian political blogging site - Pollitika.com which features a major repository of user suplied articles about politics in Croatia with more than 10 000 registered users. Each article has a tagging section where the author can supply keywords to the text, a commenting section where other users of the site can leave comments as well as a voting section where users can upvote comments of other users. To keep processing to a reasonable range, we have chosen to analyze 12 reasonably active weblogs, featuring 782 articles, 1 651 votes, and 25 592 comments. The objective of the study is to compare conceptual networks acquired using user supplied keywords (tags on articles) as well as keywords extracted through natural language processing techniques (by analyzing the text on actual articles) in order to determine which of the two approaches provides us with a better predictor of the social discourse.

The rest of the chapter is organized as follows: Firstly, in *Section 2* we give a detailed description of data retrieval. Then in *Section 3* we deal with data organization and preprocessing. In *Section 4* we provide an overview of natural language processing and provide a tool for analyzing Croatian texts. In *Section 5* we visualize two social and four conceptual networks, which were acquired through the previous sections. In *Section 6* we draw our conclusions and give guidelines for future work.

2. Retrieving Online Data

The whole process of deducing information from big sets of data necessarily starts by ways of retrieving data needed for the analyses. Such data can be found in abundance online, where almost every document, including mostly web-pages, can be retrieved and analyzed, making the Internet the most resourceful publicly available data resource at the moment. Although web sites and online documents are often under some kind of protection, which makes the data retrieval process cumbersome, tools available to the wide community of average Internet users are capable of retrieving data from a large number of online documents.

This part of the chapter is going to explain a simple process of retrieving data from online sources and preparing the collected data for further analysis. Meanwhile, some background theory will be provided, supporting the practical part of the chapter. In the end, one will be capable of creating a simple web crawling and web mining process, based on the example provided.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

2.1. Online Documents

Mentioning online documents in this chapter, one is advised to consider this term as describing anything available online, but first and foremost a simple or more complex web page. Since any web page is built using programming code, it is legitimate addressing it as an online, or web document. As such, every web page is comprised of some content, and all this content must be written down somewhere. In order for it to be written, one must use a language suitable for the given subject.

While on the subject of web documents, it is worthwhile to mention that the language standardly used to create web pages is HTML (HyperText Markup Language). Used to describe this type of semi-structured data, a markup language in general consists of markup tags. HTML documents are built using HTML tags, which, in turn, describe different document content. Every web page, even the simplest one, has the following structure, where `body` tag represents part of code which contains most of data visible to the end-user. Every tag consists of at least two parts: opening e.g. `<head>` and closing `</head>`. Any addition, such as content in between the mentioned two parts, is not obligatory. A very simple example is in code excerpt 2.1.

```
<!DOCTYPE html>
<html>
	<head>
		<title>Page Title</title>
	</head>
	<body>
		<p>This is a paragraph, part of the main content.</p>
	</body>
</html>
```

Code excerpt 2.1. The simplest HTML example

Several of these HTML tags support more specific description using attributes, as shown in code excerpt 2.2.:

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

```
<tagname attributeName=”attributeValue”>content</tagname>
```

Code excerpt 2.2. Name and value of an attribute of an HTML tag

Most commonly used attributes describing tagged content in more detail are `id` and `class`, representing unique identification for a tag, and specifying an object class mostly used by design components, respectively. Basic elements of an article of the Pollitika[1] web portal, which will be used in our analysis, are HTML tags `h1` and `p` denoting title and one content paragraph of an article respectively. An example of this is shown in **Figure 2.1**, where article titled *Manifest* starts stating "*Vrlo često kada se probudim [...]*".

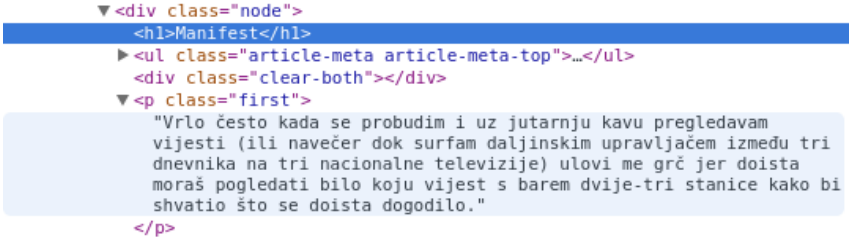


Figure 2.1.
HTML example of a part of an article where article title is highlighted dark blue, and one content paragraph is highlighted light blue

In the above example the actual data which is of interest to us is enclosed into the HTML tags. These tags, in a way, represent nodes of data that can be accessed independently, as will be shown later.

Code representation of a web page content, as seen in **Figure 2.1,** can be discovered using a regular web browser software. Covering two most widely used web browsers, Google Chrome and Mozilla Firefox, a user can read web page code by right-clicking content of interest and selecting Inspect element from the menu. A newly opened view allows the user to inspect web page code directly, which is a feature very useful when speaking of web mining.

[1] http://pollitika.com/

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

2.2. Data Retrieval Tools

Numerous tools are available Internet-wide which can be used to support the processes of web crawling (i.e. automated web page selection) and web mining (i.e. automated online data retrieval). Most of them use similar techniques, but provide users with quite different user experiences. A notable balance between those two, which further adds intuitive graphic user interface and ease of use, is the tool we chose to use in this chapter, called RapidMiner.[2]

RapidMiner is a tool which started development in 2001 and is one of the easiest to use analytics platforms usable for web crawling and mining. In addition to the ease of use it provides, technique it uses for the processes mentioned are widely used in the field, namely XPath, RegEx and similar, which we shall introduce later. Allowing process creation through element usage similar to a workflow model, RapidMiner is suitable for modular modeling and beginners, since activities needed for a full process are divided in smaller modules, which can be connected in a way suitable for the user. One of the prerequisites of using RapidMiner as a web crawling and web mining tool is installed RapidMiner extension Web Mining Extension. RapidMiner extensions can be found under Help menu, option Updates and Extensions, where Web Mining Extension is listed as one of the top downloads.

2.3. Data Retrieval Example

In this subsection, a collection of raw data is going to be retrieved from a Croatian blogging site concerned with politics, political views and user-provided perspective of the current political situation in and of Croatia, Pollitika, available at web address http://pollitika.com/.

Inputs of the web crawling and web mining process are going to be addresses of 12 user blogs hosted at the web portal. The whole process is going to perform web crawling, i.e. collecting web pages of interest based on rules provided, and web mining, i.e. extracting and retrieving interesting relevant pieces of data. For the sake of our study, we have decided to collect the following pieces of data:

- URLs, e.g. the addresses so we know where a certain piece of data has come from;
- authors of both articles and comments since they are the relevant actors of the social system to be analyzed;

[2] Available at: https://rapidminer.com/

ISSN 2285 – 4916
ISSN-L 2285 - 4916

- titles of articles and comments as well as their content to be able to analyze them using natural language processing techniques;
- metadata necessary for processing, e.g. article ID number and comment ID number;
- date and time of posting to have a temporal perspective of the collected data;
- tags (user supplied keywords) on articles to construct conceptual networks of the underlying text;
- votes, interaction and karma[3] to additionally analyze the social structure.

The final stage of the process is going to be initial data cleansing, i.e. removal or replacement of unwanted elements contained in the gathered data. At the very end, the collected dataset is saved in CSV (Comma Separated Value) file format which is very simple and suitable for further processing in various common tools. The final output will look similar to the following excerpt:

```
URL,ArticleID,ArticleType,Interaction,Karma,Tag,Title,Time,Date,ID,Author,Content

http://pollitika.com/kolinda-protiv-tita#comments,14091,Article,248,22,Kolinda
Tito,Kolinda protiv Tita.,01:42:00 PM,19.03.15 00:00,14091,Papar,"Razmišljam, ako je
nekom …"

http://pollitika.com/kolinda-protiv-tita#comments,14091,Comment,3,3,,Tvrtko
Jakovina,05:52:00 PM,09.04.15 00:00,487142,Papar,"70 godina nakon što je porazila
fašizam, ...."

http://pollitika.com/node/14091/who_voted,14091,Vote,,,,,06:22:00 PM,10.04.15 00:00,,Son
Of A Sam,
```

Code excerpt 2.3. Final CSV file excerpt

CSV files are files which store tabular data in plain-text format, meaning there are only sequences of characters. These characters are divided by a special character, most commonly used comma (,) or semicolon (;), acting as cell dividers, while rows represent table rows. As one can see in code excerpt 2.3., the first line (given in bold for sake of clarity) is the header of the document where the field types are specified separated by commas (","), while each subsequent line is an entry in this table representing a particular article, comment or vote that has been collected from the blogging site (the content of the articles and comments has been shortened in the example since entries can be very long pieces of text). The first line is an article entry, the second a comment and the third a vote.

[3] Interaction and karma are statistics generated by the blogging site based on user behavior and interaction.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

2.3.1. Process Overview

Every RapidMiner process consists of one or more operators, where most operators have mandatory input streams, although some arguably simpler arguments exist, which need no input to be provided. A number of operators is provided for a user to use by default, but even more are available when using the *Help* menu, and selecting *Updates and Extensions*. Most operators represent activities on their basic level, with no further operators being part of them. These are simple operators which will be called operators in this chapter. More complex operators are those who are divisible, i.e. they are made of several operators, iterating over them. These complex operators are called subprocesses, since they represent a process, but are already included in a higher-level process. Therefore, we shall use term **operator** for non-divisible, single-leveled operators, and term **subprocess** for operators containing operators within themselves, i.e. multi-level operators.

Each operator or subprocess used in RapidMiner must be customized using operator parameters which allow for basic operator instructions or fine tuning operator behaviour, an example of which is shown on **Figure 2.2.** Exceptions are some flow-merging operators, which have no parameters to be specified.

Every operator or subprocess used in a RapidMiner main process can be renamed for the sake of clarity. In this chapter, when naming a certain operator or subprocess, we shall state that a custom named operator is of type of RapidMiner named operator, e.g. article extraction is conducted by Get Article operator (which is) of type `Cut Document`; where `Get Article` is clearly the custom name and `Cut Document` is an internal RapidMiner name.

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

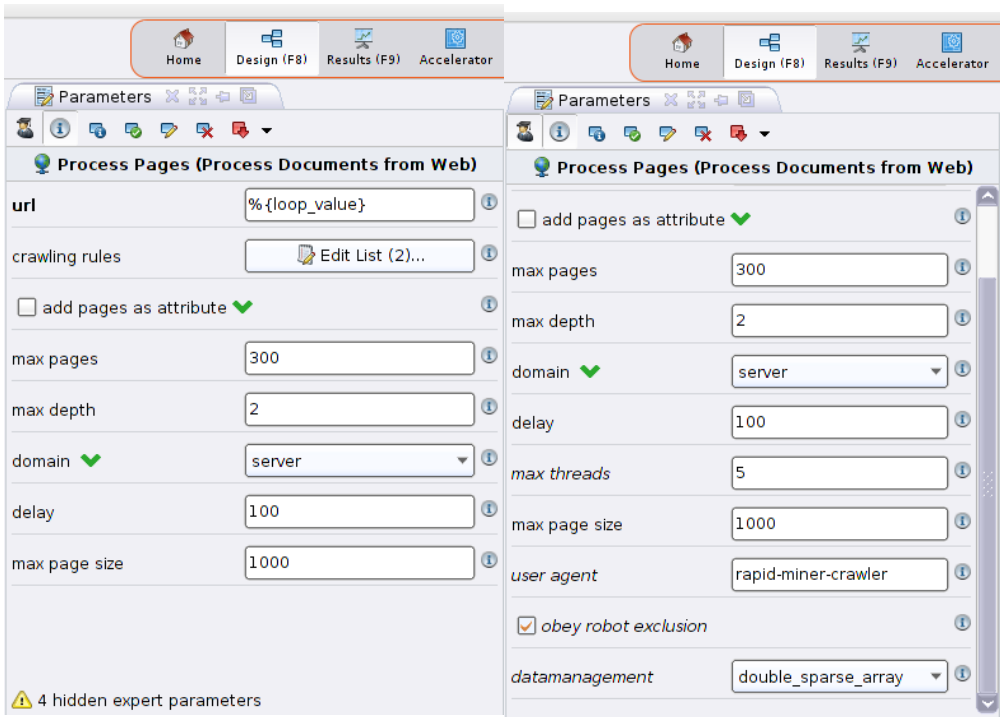


Figure 2.2.
Parameters of operator of type Process Documents from Web; four parameters are hidden (left), only visible in expert mode enabled (by pressing key F4 on the keyboard),shown in italics (right)

The example process in this chapter consists of three operators and two subprocesses, shown on **Figure 2.3.** Connected together they produce output as mentioned above. Those are as follows:

- `Read Links` operator of type `Read CSV` is used for importing and reading a CSV file with a list of URLs, i.e. links to user blogs hosted on the Pollitika portal; the best way of setting up this operator is using the `Import Configuration Wizard` since it allows for a simplified process of specifying operator parameters' values.
- `Loop Links` subprocess of type `Loop Values` is used to iterate through the list of links provided by `Read Links` operator. This subprocess consists of second-level operators we shall describe later in the chapter. Looping subprocesses, such as `Loop Values,` asks for one rather important parameter: `iteration macro.` Value of this parameter is used as a reference to the data available by current loop iteration, to be used inside the subprocess. This means that every operator or subprocess contained in this subprocess can retrieve data by simple usage of the named macro.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

- For example, let one of the input links be http://pollitika.com/blog/starpil, and let value of the parameter `iteration macro` be `loop_value`, then an operator contained in subprocess `Loop Links` can retrieve stated link by using macro `%{loop_value}` as value of a relevant parameter instead of stating the actual link. This feature is useful because values of a macro change with every iteration of a subprocess, and this allows its usage in lower levels of a subprocess.

- `Merge Results` operator of type `Append` is here used as a collector of all the data generated by `Loop Links` subprocess. In general, `Append` operator adds all input examples into a combined set. This particular `Append` operator merges all data retrieved from the Web.
- `Clean Results` subprocess of type `Subprocess` contains operators used for data modification, since some impurities are collected along with the usable data. Excessive spaces, unimportant HTML tags and similar are either deleted or replaced by operators of this subprocess. Subprocess of type `Subprocess` is used in general only to provide a shell for a subprocess, and contains no additional parameters.
- `Write Results` operator of type `Write CSV` must have a parameter *csv file* provided with resulting CSV filename and location. This operator serves as a save file command, used to save processed data to a file, and thus enable simple further use.

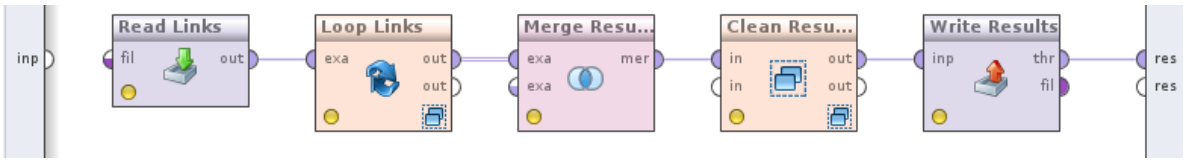


Figure 2.3.
Web crawling and web mining process overview containing three operators and two subprocesses (identified by small icon in lower right corner of an element)

As mentioned above, in the beginning of the process, a list of URLs, i.e. links to user blogs hosted on the Pollitika portal, must be imported and read. Operator `Read Links` of type `Read CSV` is charged with this task, with user defined location and name of the needed file as parameter `csv file`, and other parameters which are not of interest at the moment, since only one link is placed in every row of the document, as shown in next code excerpt:

ISSN 2285 – 4916
ISSN-L 2285 - 4916

```
Link

http://pollitika.com/blog/starpil

http://pollitika.com/blog/papar
```

Code excerpt 2.4. CSV file links list excerpt

CSV file such as the one with excerpt shown above, can be thought of as a simple table with only one column and several rows where every table row contains one link.

In brief, `Loop Links` subprocess reads every line of the provided links list and processes it further looping through all the available links provided by `Read Links` operator. Every iteration of this subprocess creates some results, which are then collected by the `Merge Results` operator. After the `Loop Links` subprocess has processed all the needed data, by performing all the needed iterations, the `Merge Results` operator merges all the data in one dataset ready to be passed on to the next operator. Provided data needs to be cleansed of impurities and uninteresting elements for the oncoming analyses using subprocess `Clean Results`. The final dataset given as output of the cleaning subprocess is then saved in a CSV file by the operator `Write Results`, for improved convenience and ease of further analysing the collected data.

The easiest way of setting up the first activity of importing list of links is using the `Import Configuration Wizard` since it will ask about file name, location, delimiter characters and data type saved in the file.

2.3.2. Web Crawling

The first level of the `Loop Links` subprocess, shown on **Figure 2.4**, consists of another subprocess named `Process Pages`, of type `Process Documents from Web`. Input to this subprocess is one link per iteration of aforementioned `Loop Links` subprocess. Since this input is dependent on iteration performed by `Loop Links` subprocess, no definite link is given as a value of `Process Pages'` parameter `url`.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Figure 2.4.
Web crawling level

In order to reference the link of every iteration, we use a macro statement. Using parameter `iteration macro` of `Loop Links` subprocess we name it, e.g. `loop_value`, and using an expression, e.g. `%{loop_value}`, we use it with `url` parameter of Process Pages. This means that `Process Pages` references value iterated by `Loop Links`.

`Process Pages` subprocess is used for automated web navigation and storing pages along with additional extracted information. Web navigation starts on a web page with URL stated in `url` parameter, while navigation is defined by the `crawling rules` parameter. In addition to simple navigation regulated by follow link rules, user can specify which visited web pages are of interest, and should be saved, using store rules. List parameter `crawling rules` may contain a set of one or more rules. These rules are applied as: following rule with matching url, following rule with matching text, storing rule with matching url, and storing rule with matching content. Following rules are used for navigation, while storing rules are used for saving web pages; defining if links are to be followed based on their URL or text, and if pages are to be saved based on their URL or content respectively. Each rule is evaluated against a `rule value`. `Rule values` are most often expressed using regular expression.

Regular expression (RegEx) is a series of characters which form a pattern against which is then some text valuated and deemed similar or dissimilar. RegEx is mainly used for pattern matching with strings, e.g. for functions such as *find* or *find and replace*. In the context of `Process Pages` subprocess RegEx is used as a pattern for recognition of links which are to be followed or saved for further processing. Basic RegEx statements are enumerated in **Table 2.1**. For each RegEx statement description is provided. It would help noting that RegEx statements form patterns, therefore as such are not characters but represent one or more characters.

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Table 2.1. Basic RegEx elements

RegEx	Description
.	any character except line break
\w \d \s	word, digit, whitespace
[cro]	any of the characters *c*, *r* or *o* (one character only)
\, \. \;	comma, period, semicolon
w* w+ w?	0 or more, 1 or more, 0 or 1 character repetition of character *w*
w{3} w{2,}	exactly 3, 2 or more
w|y	w or y

Several simple RegEx combinations and examples are shown and described in **Table 2.2**.

Table 2.2. RegEx examples and statement combinations

RegEx statement	Input	Match
[no]	RegEx example in year 2015 on 25/4. With no mistake!	o; n; n; o
Pattern matches any appearance of characters n or o. Only one letter is matched.		
[A-Z]\w*	RegEx example in year 2015 on 25/4. With no mistake!	RegEx; Ex; With
Pattern matches words starting with a capital letter. Since only letters are included in the pattern, breaking points are all non-letter characters, e.g. whitespace, numbers etc.		

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

\d+\/\d	RegEx example in year 2015 on 25/4. With no mistake!	25/4
Pattern matches one or more digits followed by a forward slash (/), followed by a digit. Special characters such as forward slash, period, colon etc. have to be preceded by a backslash if they are to be matched literally.		
n\s(\w+)\s\d{2,}	RegEx example in year 2015 on 25/4. With no mistake!	year
Only pattern in brackets is returned, as other elements are used for further specifying the pattern. Match consists of letter n followed by whitespace (\s) followed by one or more letters, then whitespace again and at least two digits. Only the word in the middle is returned because only that part of the pattern is bracketed.		

After choosing what kind or rule is going to be used in an entry, it is necessary to enter the value, which is to be used, using regular expression statements. The example in this chapter follows links containing word "page=" or ending with word "comments" or containing the word "node" followed by numbers followed by the word "who_voted". Creation of a RegEx statement matching the described structure is built step-by-step in **Table 2.3**. Similarly, web pages which are to be saved are regulated by a storing rule, namely `store_with_matching_url,` which evaluates a web page URL against rule value. Short analysis of the Pollitika.com portal which is used as a source of data for this chapter resulted in a simple rule concerning which pages contain article data, which pages contain article data accessible through pagination, which pages contain relevant comments of an article, and which pages contain voting information. For example, every article is identified by a unique number. The page containing voting information concerning article with `ID 14152` is displayed on a page with the URL http://pollitika.com/node/14152/who_voted. Therefore, when gathering voting information, we are interested in web pages containing a numerical sequence followed by the word "who_voted". Similarly other statements were formed, and in the end whether a web page is saved or not is determined by a RegEx statement `.+page\=.+|.+\#comments|.+\/who_voted` meaning we are interested in web pages with URL containing word "page=" or ending with words "#comments" or "/who_voted".

ISSN 2285 – 4916
ISSN-L 2285 - 4916

Table 2.3. RegEx statement step-by-step construction

RegEx	Description
page\=	We are looking for word "page" immediately followed by equal sign: "page="
.+page\=.+	Word "page=" must have one or more characters (excluding line breaks) preceding it, and one or more characters following it
#comments	We are looking for word "comments" immediately preceded by character "#": "#comments"
.+\#comments	Word "#comments" must have one or more characters preceding it, but must not be followed by anything
\/node\/	We are looking for word "/node/"
\/who_voted	We are looking for word "/who_voted"
.+\/node\/\d+\/who_voted	We are looking for a combination of 1 or more characters (.+), word "/node/", 1 or more digits(\d+), and word "/who_voted" followed by nothing
.+page\=.+| .+\#comments| .+\/node\/\d+\/who_voted	In the end, we are creating a combination of all three statements divided by an OR statement (denoted with the dash "|" symbol) meaning the pattern we are looking for either contains word "page=" preceded and followed by one or more characters, or contains word "#comments" preceded by one or more characters and followed by nothing, or it must be a sequence of 1 or more characters (.+), word "/node/", 1 or more digits(\d+), and word "/who_voted" followed by nothing

Some other parameters, in addition to `url`, can be set when working with a `Process Documents from Web` type of subprocess. These include `max pages` (maximum number of pages to be visited by the subprocess), `domain` (if value "server" is set, all the followed pages will be on the same server, i.e. links are starting with pollitika.com), `max page size` (constraints size of saved web page in kB). All of these can

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

be customized, according to a given situation. Even so, some standard values exist for these parameters. `max pages` parameter is important when a user determines how much data is needed in the end. Higher value allows for more pages to be discovered or visited, while smaller numbers can be used for training or testing purposes, only to inspect success of some other operators. `max depth` decides on the number of levels which will be visited, as shown in **Figure 2.5.** Some web pages experience issues with their systems, or the server is not adequate for the workload it received. These and many more time-related problems can be solved using delay parameter which contains time in milliseconds before an action is performed after following a certain link. The most important element needed for quality values in these parameters is careful planning of web pages visited. For the sake of this experiment, we set parameters as follows: `max pages` to 300, max depth to 2, `domain` to server, `delay` to 100, `max page size` to 1000.

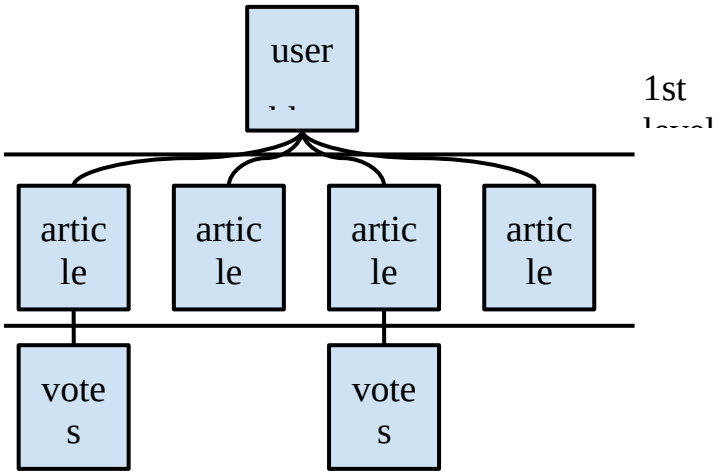


Figure 2.5.
Simple depiction of relevant Pollitika.com structure

2.3.3. Web Mining

Web mining in RapidMiner relies heavily on usage of information extraction using the `Extract Information` operator. All instances of this operator are renamed in the example process for improved clarity. Flow on this level of the process is visible on **Figure 2.6.**

All the operators and subprocesses visible on **Figure 2.6** are located inside `Process Pages` subprocess of type `Process Documents from Web.` As such, these operators and subprocesses work with data provided by `Process Pages` subprocess consisting of a number of saved relevant web pages.

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

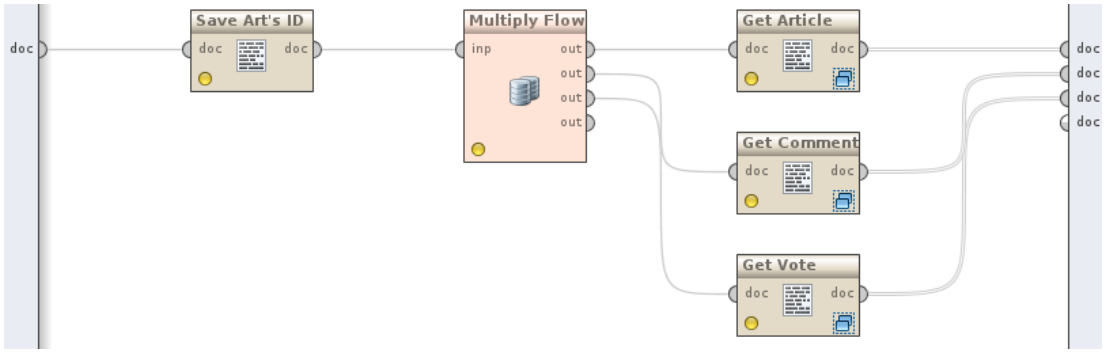


Figure 2.6.
Web mining level

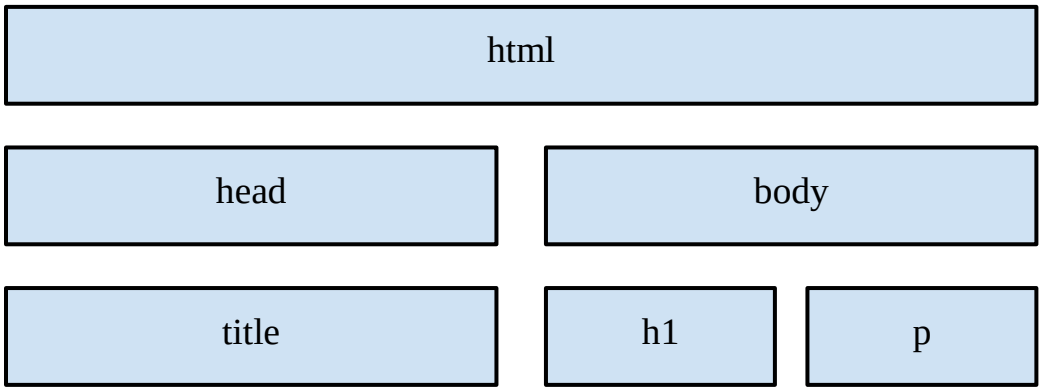
We first add an operator, `Save Art's ID` which is of type `Extract Information`, that extracts article ID using `query type` Regular Expression. Other available query types are: string matching, regular region, XPath, indexed and JsonPath. These might be useful for exactly matching a string, setting borders of interesting content using RegEx, or using XPath. Just as explained in an example above, article ID is stored in one link somewhere on the web page. It would be much more difficult if we were unable to set a RegEx statement which identifies precisely the piece of information needed. RegEx we use in `regular expression query` parameter is `\/(\d+)/who_voted` saving the extracted data in attribute named `RefID`. This pattern matches that first occurrence of text matching pattern: numbers + "/who_voted", is examined and only the part containing numbers (RegEx `\d`) is saved as attribute value. Further entries could be added to parameter list regular expression queries.

Data flow is then multiplied by the `Multiply Flow` operator. Three operators of `Cut Document` type representing subprocesses: `Get Article`, `Get Comment` and `Get Vote`, are used as noted in their title - for extracting article, comment or vote respectively. Each of these subprocesses uses XPath queries to return the specified part of a web page.

XPath expressions are used to navigate through elements and attributes in an XML, and, by extension, HTML documents. Building XPath expressions is not as complex as it seems at first sight. Even though a user can write an XPath expression by themselves, valuable assistance can be gained from web browser usage. Using either of the most popular web browsers, `Inspect element` ability is a welcome ally when we are defining

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

an XPath expression. Right-clicking on an interesting piece of content and choosing `Inspect element` from the context menu, a developer's tool is shown, and the selected web page element is highlighted. From it, e.g. in Google Chrome, one is free to right-click on a content element of interest and choose `Copy XPath` from the context menu. This command copies to clipboard XPath expression leading to the selected content element, i.e. part of a web page. XPath expression represents a kind of an address leading to the selected content element. Visual example is given in **Figure 2.7** where structure shown in code earlier in this chapter (code excerpt 2.1.) is visualized. Looking at **Figure 2.7**, it is clearly visible that body tag can be considered a child of html element, and, in turn, h1 and p can be considered children of body element. Two children elements with the same parent element are called siblings, similar to a simple family tree. One element being a child of another one means that the subject element is a part of the parent element. In other words, every parent consists of child elements. Inversely, child elements are building blocks of parent elements. For example, body element from **Figure 2.7** is a child of html, but it is a parent of h1 and p. Child-parent, being one of the most important concepts in XPath, is written as follows: html/body; denoting that html tag is a parent of body element. Supporting simpler notation, and easier navigation, XPath allows for non-direct path expression, e.g. html//h1, meaning that we aim to select all h1 elements which are descendants (children of children of children...) of html element. When trying to address parent element, one should use "..", which is an element for path advancement to a higher level, e.g. //h1/../p is an XPath expression which starts from any h1 element, follows the way to the parent, then descends again top element. Other than addressing tag elements as themselves, XPath can help path selection process by selecting content elements with an addition of an attribute. For example, p HTML tag containing attribute named ID, can be referenced as //p[@id="1123"]; the system is going to find a paragraph element with ID value of 1123.



ISSN 2285 – 4916
ISSN-L 2285 - 4916

Figure 2.7.
Visualization of a simple HTML document structure

`Get Article` subprocess of type `Cut Document` is used to extract a larger piece of data from a document and run it through a process. Part of a document to be extracted is defined using several different query types identical to those in `Extract Information` operator description. In this example process we defined `a`ll the wanted data set extraction rules for articles, comments and votes in a similar way, using XPath expressions. XPath expressions such as the one for article extraction was `gathered` and copied by using `Inspect Element` tools. Simple right click on an interesting element and choosing Copy `XPath` command provides us with a simple way of creating an XPath expression such as `//*[@id="content-main"]/div[3]/h1` which leads to the title of an article. The mentioned XPath expression looks for any HTML element with attribute of name `id` and value `content-main`, containing a `ul` tag whose parent contains `div` tag with attribute of name `class` and value `node`.

Expressions used as XPath queries in these three subprocesses (`Get Article, Get Comments, Get Vote`) are generated semi-automatically, one part by using `Inspect element` command of a web browser, the other part being direct user intervention. Resulting XPath expressions are as follows:

- Article: //*[@id="content-main"]/h:ul/../h:div[@class="node"]
- Comment: //h:div[contains(concat(' ', @class, ' '), ' comment-published ')]
- Vote: //h:tbody/h:tr

Prefix `h:` is commonly used in RapidMiner XPath expressions, because it denotes the HTML namespace. Therefore every HTML tag mentioned in an XPath expression[4] must be preceded by `h:`. This extracted element of the web page is then forwarded to the lower level, where two operators extract useful data (e.g. `Title, Tag, Date, Author,` and other attributes). Extraction rules for the mentioned attributes intensively depend on XPath expressions and RegEx statements. While XPath expressions are localized, to be valid inside extracted article, comment or vote, since many elements could be considered duplicates on

[4] Some of the more complex elements of XPath expression are not covered by this chapter for the sake of brevity but the interested reader is advised to consult the appendix for additional resources.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

the whole web page otherwise, RegEx statements are not dependent on the extract, although working on a smaller area does make the process easier. Attribute values of an article are collected using the following XPath expressions:

- Title - //h:h1/text()
- Karma - //h:span[@class="karma_score"]/text()
- Interaction - //h:span[@class="broj-komentara"]/text()
- Tag - //h:div[@class="taxonomy"]/h:ul
    - and the following RegEx statements:
- Author - ([\wšđćžčŠĐČĆŽ!_?\.\s]+)\s.\s[\wČ]{3},
- Date - \d{2}\/\d{2}\/\d{4}
- Time - \d{2}\:\d{2}
- Content - (?sm) (.*)
- RefID - /comment/reply/(\d+)#comment-form
- ID - /comment/reply/(\d+)#comment-form

Let us clarify some of them. The `Title` attribute is found by a simple search for an `h1` tag and extracting its textual content. `Interaction` number is gathered from textual content of an element with class "broj-komentara". `Author` name is extracted from a pattern containing a series of characters (including any letter, special Croatian letters, exclamation mark, underscore, question mark, period, whitespace) which make the actual author name, followed by a whitespace, followed by any character, then by a whitespace and a series of three letters including letter Č. `Date` of an article or a comment is gathered as a pattern looking for two numbers, forward slash, two numbers, forward slash and four numbers to end with. Almost completely the same is the process for retrieving attribute values for comments and votes. The only difference is in query values, since every type of element of interest has different structure, e.g. articles are contained in an element of class `node`, while votes are located inside a simple table. Further difference is obvious in content of the mentioned content types, e.g. author name in an article is not located similarly to the author name in votes. Information from all of the three content types is extracted using both RegEx and XPath expressions.

After merging results from all the visited and saved web pages, those results are to be cleaned in order to minimize risk of problems in further analyses and to optimize data formatting.

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

2.3.4. Cleaning Retrieved Data

Data collected in this process contains some elements which are not interesting, hinder further analyses, or are simply redundant. Several examples are shown in **Table 2.4**. below in a state before being run through the `Clean Results` subprocess of type `Subprocess`, and after the whole cleaning process.

Table 2.4. Example of the cleaning process

Before	After
"<ul xmlns='http://www.w3.org/1999/xhtml' class='links inline'> <li class='taxonomy_term_366 first'> hrvatska politička scena <li class='taxonomy_term_3937 last'> Ivo Josipović "	"hrvatska politička scena, Ivo Josipović"
Tag attribute of an article collected a lot of unwanted elements while gathering useful data. Therefore cleaning is required. Cleaning process for Tag attribute requires removal of HTML tags, removal of excess spaces and addition of commas between two different tags. Resulting data is nicely formatted, simply delimited by a comma and much more memory-efficient.	

The cleaning process starts by removing duplicates, based on the ID attribute of collected content which contains values uniquely identifying every instance of article and comment. Next step is removing all varieties of quotation marks and replacing them with single quotation marks, i.e. apostrophes (‘), as shown on **Figure 2.8**.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

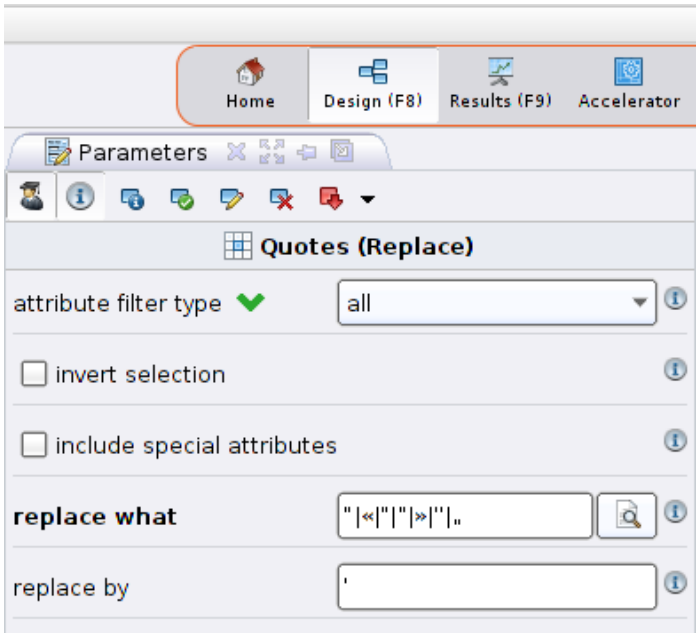


Figure 2.8.
Parameters of Quotes operator of type Replace for replacing all variations of quotation marks ("|«|"|"|»|''|„)

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

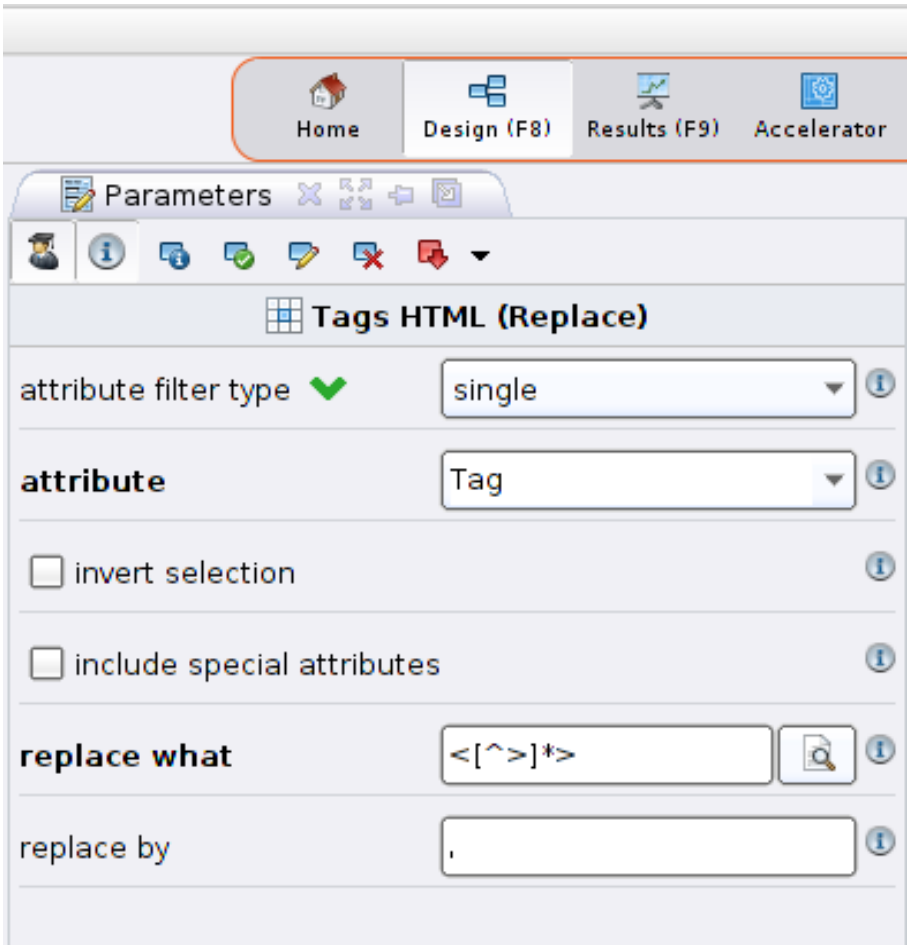


Figure 2.9.
Parameters of Tags HTML operator of type Replace for replacing all HTML tags from Tag attribute, RegEx <[^>]*>

During the collection process, some of the HTML tags were collected along with the interesting data. Therefore, HTML tags have to be removed from the collected dataset. First, HTML tags are removed from the Tag attribute and comma delimiter is added, as depicted by **Figure 2.9**. Secondly, HTML tags are removed from all other attributes. Lastly, excessive whitespace in form of successive spaces and newline elements, along with whitespace at the beginning and end of attribute value, is removed. All of the replacement done is stated in **Table 2.5**.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Table 2.5. All replacements and removing actions included in cleaning subprocess

Replace what	Replace by	Description
"|«|"|"|»|''|„	'	Removing all varieties of double quotation marks
<[^>]*>	,	Removing HTML tags from Tag attribute and replacing them with commas
[,\s]{2,}	,	Replacing multiple successive comma-whitespace combinations with single comma
^,		Removing comma from the beginning of a table cell
, $		Removing comma from the end of a table cell
<[^>]*>		Removing HTML tags from all attributes and replacing them with whitespace
\s{2,}		Replacing multiple successive whitespace with single whitespace
^\s		Removing whitespace from the beginning of a table cell
\s$		Removing whitespace from the end of a table cell

In the end, cleansed data is saved in CSV file format, ready to be used, excerpt of which follows:

```
"Tag";"URL";"ArticleID";"ArticleType";"Interaction";"Karma";"Title";"Time";"Date";"ID";"Author
";"Content"
```

ISSN 2285 – 4916
ISSN-L 2285 - 4916

```
"hrvatska politička scena, Ivo Josipović";"http://pollitika.com/josipovicevi-mrtvi-
kapitali#comments";"14124";"Article";"10";"15";"Josipovićevi mrtvi
kapitali";"18:16";"07/04/2015";"14124";"StarPil";"Ivo Josipović, bivši predsjednik RH, u vrlo
kratkom vremenu prosuo je sve ono što je mislio da je skupio.[...]"

;"http://pollitika.com/josipovicevi-mrtvi-kapitali#comments";"14124";"Comment";"1";"1";"nikako
se ne slažem s tobom";"21:19";"09/04/2015";"487146";"fuminanti";"i predsjednik i premjer su
barem u početnoj fazi radili upravo ono što sam od njih htjela:[...]"
```

Code excerpt 2.5. An example of retrieved and clean data

2.3.5. Process files

The whole process is available for download, pointed to in Appendix to this chapter. Process file is ready to be imported to RapidMiner using Import Process option. List of links used in the process described above is available for download, again pointed to in Appendix to this chapter.

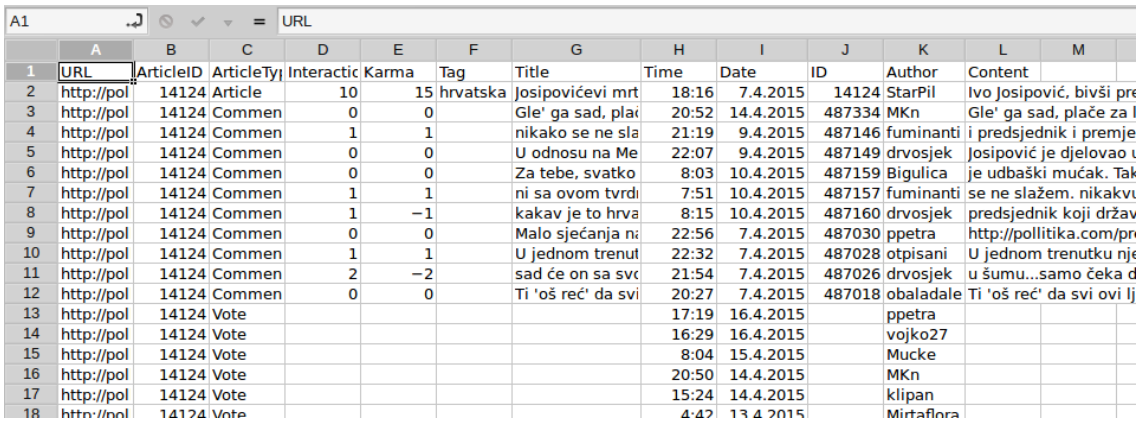
3. Data Organization and Preprocessing

After the data has been collected in a CSV file, as described above, the retrieved dataset has to be preprocessed and organized in order to be usable by various analysis and visualization tools. Herein we will use a simple spreadsheet program[5] to conduct this task since CSV files can be easily imported into such programs.

As we saw in the previous section, the output CSV file is a huge table in which each row represents either an article or a comment or a vote (see **Figure 3.1**). Thus, the first thing we have done is to separate these three distinct categories of data into three individual sheets.

[5] We need to comment on this approach beforehand: using spreadsheet programs on very big datasets is often unfeasible since computation and inference time rises heavily with the amount of data. There are alternatives including using cloud based services like Google's Big Query or by implementing a custom preprocessing script. Since such services aren't free and since programming a custom script is out of the scope of this chapter, we decided to use spreadsheet programs which can handle reasonably big datasets. Herein we will use the free spreadsheet program Gnumeric (available at http://www.gnumeric.org/) due to its speed, but more widespread alternatives like Microsoft Excell or LibreOffice Calc can be used in the same fashion as well, without or eventually minimal intervention.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	URL	ArticleID	ArticleTyp	Interactic	Karma	Tag	Title	Time	Date	ID	Author	Content	
2	http://pol	14124	Article	10	15	hrvatska	Josipovićevi mrt	18:16	7.4.2015	14124	StarPil	Ivo Josipović, bivši pre	
3	http://pol	14124	Commen	0	0		Gle' ga sad, plać	20:52	14.4.2015	487334	MKn	Gle' ga sad, plače za l	
4	http://pol	14124	Commen	1	1		nikako se ne sla	21:19	9.4.2015	487146	fuminanti	i predsjednik i premje	
5	http://pol	14124	Commen	0	0		U odnosu na Me	22:07	9.4.2015	487149	drvosjek	Josipović je djelovao u	
6	http://pol	14124	Commen	0	0		Za tebe, svatko	8:03	10.4.2015	487159	Bigulica	je udbaški mućak. Tak	
7	http://pol	14124	Commen	1	1		ni sa ovom tvrdn	7:51	10.4.2015	487157	fuminanti	se ne slažem. nikakvu	
8	http://pol	14124	Commen	1	−1		kakav je to hrva	8:15	10.4.2015	487160	drvosjek	predsjednik koji držav	
9	http://pol	14124	Commen	0	0		Malo sjećanja na	22:56	7.4.2015	487030	ppetra	http://pollitika.com/pr	
10	http://pol	14124	Commen	1	1		U jednom trenut	22:32	7.4.2015	487028	otpisani	U jednom trenutku nje	
11	http://pol	14124	Commen	2	−2		sad će on sa svo	21:54	7.4.2015	487026	drvosjek	u šumu...samo čeka d	
12	http://pol	14124	Commen	0	0		Ti 'oš reć' da svi	20:27	7.4.2015	487018	obaladale	Ti 'oš reć' da svi ovi lj	
13	http://pol	14124	Vote					17:19	16.4.2015		ppetra		
14	http://pol	14124	Vote					16:29	16.4.2015		vojko27		
15	http://pol	14124	Vote					8:04	15.4.2015		Mucke		
16	http://pol	14124	Vote					20:50	14.4.2015		MKn		
17	http://pol	14124	Vote					15:24	14.4.2015		klipan		
18	http://pol	14124	Vote					4:42	13.4.2015		Mirtaflora		

Figure 3.1.
Initial dataset opened in a spreadsheet program

To do this one can use the *autofilter* option or advanced filters, which most spreadsheet programs have implemented. In our case we have created three separate sheets entitled: *articles-clean*, *comments-clean* and *votes-clean*.

One of the tasks we set out in this study were to analyze the social and conceptual networks of the blogging site at hand. From the acquired dataset we can construct at least two social networks: (1) the commenter networks (e.g. two users are connected if one has commented on the other's article), and (2) the voter network (e.g. two users are connected if one has voted for the other's comment).

To construct the first network, we had to create a new sheet in our dataset and list the individual username pairs of users who were authors of articles and users who commented on exactly these articles. Since the *comments-clean* sheet contains only the Article ID of the article on which the comment was made, but not the author, we needed to connect the sheet with the *articles-clean* sheet. To do that, we used two formulae:

```
=vlookup('comments-clean'!B2;'articles-clean'!$J$2:$K$32000;2;0)

='comments-clean'!K2
```

Code excerpt 3.1. Functions for constructing the commenter network

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

The first formula uses the *vlookup* function which allows us to find a given value in another table. It takes 4 parameters: (1) the value to be found (in our case the Article ID), (2) the range where the value is to be found (in our case a data range of the *articles-clean* sheet that includes Article IDs and Authors), (3) the index of the to be returned value (in our case for an Article ID we wanted to return the Author's username, which is the second column in the range), and (4) an optional parameter is the search range in (2) is to be treated as sorted or not (in our case the dataset wasn't sorted). The second formula just finds the username of the commenter from the *comments-clean* sheet. The result is a sheet similar to the one on **Figure 3.2**.

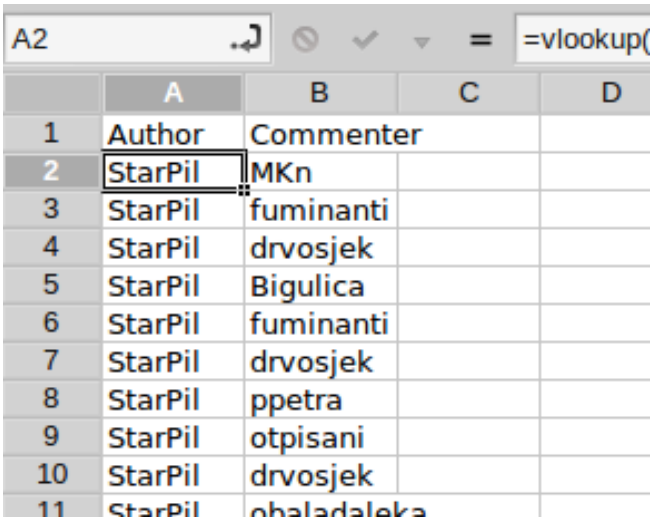


Figure 3.2.
Sheet containing the commenter network

Analogous, the voter network is constructed using the formulae:

```
=vlookup('votes-clean'!B2;'articles-clean'!$J$2:$K$32000;2;0)
```

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

```
=='votes-clean'!K2
```

Code excerpt 3.1. Functions for constructing the commenter network

In order to construct conceptual networks one needs to extract concepts on one hand, and connect concepts into a network, on the other. The former, in our case, can be done using two different techniques: (1) by using user supplied keywords (e.g. tags on articles) or (2) by using natural language processing techniques to automatically extract keywords from text. The user supplied tags have already been extracted in the previous section, and we shall deal with the second approach in the following section. Regarding the establishment of links between concepts one can use various approaches. One interesting approach is to construct a *folksonomy* (Mika, 2007, but see also Schatten et al. 2011; Schatten, 2013 as well as Pejić Bach et al., 2013 for a slightly different approach). A *folksonomy* (similarly to a more formal *taxonomy*) is a certain mathematical structure (an n-partite hypergraph) in a special context (social applications). In our case if we take that hypernodes are the triples (author, article, keyword) in which are connected with other hypernodes through mutual authors, articles or keywords we get a structure (a 3-partite hypergraph) which reflects the conceptual and social network of a given social application.

By using a special mathematical technique called graph folding it is possible to construct special bipartite graphs out of this structure, some which allow us to create conceptual networks. In our case there are at least two such conceptual networks constructed by using the following simple rules: (1) two concepts (keywords) are connected if they have been used on the same article, and (2) two concepts are connected if they have been used by the same user. These two types of conceptual networks often have quite different characteristics: while the former often connects semantically similar concepts (intuitively if two concepts have been used to describe the same text, they are probably semantically similar), the latter often reflect the interests and internal conceptual networks of users (intuitively if two concepts have been used by the sam user, the user might have an interest in both). It is also possible to construct other types of conceptual networks, but they are from our perspective not relevant for the sake of this study.

To construct either of these networks is not a trivial task in a spreadsheet program,[6] but we will show how it is possible here, nevertheless. Firstly, we created a new sheet *tags*-

[6] A much more versatile way would be to implement a script or use regular expressions, but such techniques are outside the scope of this chapter.

ISSN 2285 – 4916
ISSN-L 2285 - 4916

clean and added just two fields: Article ID and Tag (by referencing the *articles-clean* sheet). The Tag field is a ‘,’-separated list of keywords as provided by authors. What we want to do is to split this list into separate cells that can be processed further. Notice that there is a variable number of keywords on each article as well as that keywords are of variable lengths, which complicates the procedure. An important thing to become aware of is that the keywords are separated by a special delimiter (‘,’) which means that by searching for the delimiter one is able to split the string into particular keywords. In the following we will show hot to extract the first *n* keywords of the Tag field, but by using the same procedure all keywords could be extracted. We have chosen to use only 8 keywords for sake of simplicity.

The first thing we have done is to add an additional delimiter sign at the end of the Tag string (to be able to find the last character of the last keyword) and converted the string to lowercase so that keywords would be equal regardless of case in the consequent processing. This was achieved using the following formula:

```
=lower(concatenate(B3;","))
```

Code excerpt 3.2. Formulae for adding a delimiter and converting to lowercase

The second step was to calculate 8 numbers which represent the positions of the first 8 delimiters in the lowercased string. This was achieved using the following formulae:

```
=find(",";$C3)
=find(",";$C3;E3+1)
=find(",";$C3;F3+1)
=find(",";$C3;G3+1)
=find(",";$C3;H3+1)
=find(",";$C3;I3+1)
=find(",";$C3;J3+1)
=find(",";$C3;K3+1)
```

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Code excerpt 3.3.
Formulae for adding a delimiter and converting to lowercase

The function *find* returns the position of a given search string (the first parameter) inside another string (the second parameter), and has an optional third parameter to enter the position in the to be searched string on which to start looking for the search string. The formulae were put into consequent columns so that they reference each other, whereby *C3* is the field where the lowercased string was stored, the first formula was in column *E*, the second in column *F* etc. In case the searched string has fewer than 8 keywords, the formulae will return *#VALUE!* errors, which we will circumvent in the next step.

The resulting numbers can now be used to extract the individual keywords. This was achieved by using the following formulae:

```
=iferror(left($C3;E3-1);"")
=iferror(mid($C3;E3+2;F3-E3-2);"")
=iferror(mid($C3;F3+2;G3-F3-2);"")
=iferror(mid($C3;G3+2;H3-G3-2);"")
=iferror(mid($C3;H3+2;I3-H3-2);"")
=iferror(mid($C3;I3+2;J3-I3-2);"")
=iferror(mid($C3;J3+2;K3-J3-2);"")
=iferror(mid($C3;K3+2;L3-K3-2);"")
```

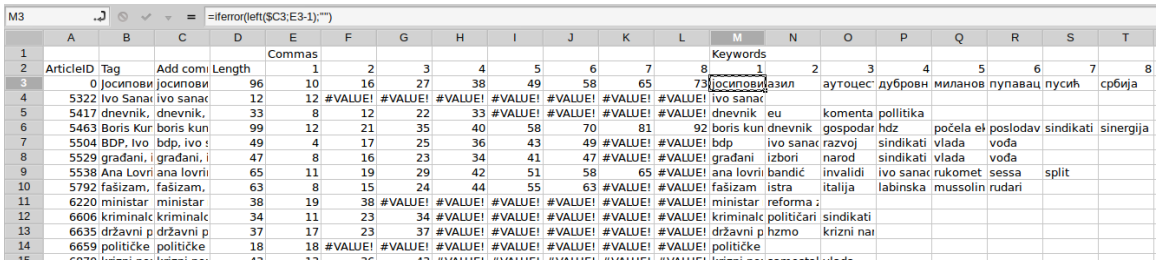
Code excerpt 3.4. Formulae for extracting the first 8 keywords

The function *iferror* allows us to get rid of the previously mentioned errors. It takes two parameters: (1) the formula to be checked for errors and (2) the value to return if there is an error. If there is no error the formula returns the result of the formula in the first parameter. The *left* function takes two parameters: (1) the string from which to extract an excerpt and (2) the number of characters to extract from the beginning (the left side) of the string. We used left to extract only the first keyword since the initial position (the start of the string or 0) is known to us. For the other keywords this position is unknown, e.g. it

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

depends on the length of the previous keywords. Thus, we used the *mid* function which takes 3 parameters: (1) again the string from which to extract, (2) the starting position of the to be extracted substring, and (3) the ending position of the excerpt. As one can see, these positions depend on the previous positions with some additional calculations to take care of the delimiter sign and whitespace that we do not want in the resulting keyword.

The resulting sheet now looks similar to the one shown on **Figure 3.3**.

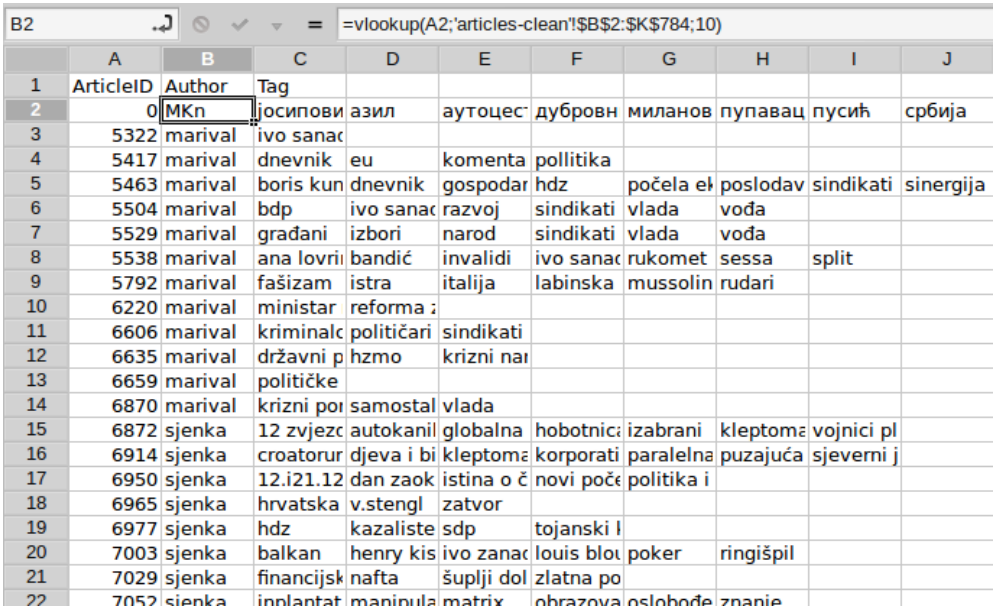


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1					Commas								Keywords							
2	ArticleID	Tag	Add comi	Length	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
3	0	Јосипови	јосипови	96	10	16	27	38	49	58	65	73	јосиповиразил		аутоцес	дубровн	миланов	пупавац	пусић	србија
4	5322	Ivo Sanac	ivo sanac	12	12	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	ivo sanac							
5	5417	dnevnik,	dnevnik,	33	8	12	22	33	#VALUE!	#VALUE!	#VALUE!	#VALUE!	dnevnik	eu	komenta	pollitika				
6	5463	Boris Kur	boris kun	99	12	21	35	40	58	70	81	92	boris kun	dnevnik	gospodar	hdz	počela el	poslodav	sindikati	sinergija
7	5504	BDP, Ivo	bdp, ivo s	49	4	17	25	36	43	49	#VALUE!	#VALUE!	bdp	ivo sanac	razvoj	sindikati	vlada	voda		
8	5529	građani, i	građani, i	47	8	16	23	34	41	47	#VALUE!	#VALUE!	građani	izbori	narod	sindikati	vlada	voda		
9	5538	Ana Lovri	ana lovri	65	11	19	29	42	51	58	65	#VALUE!	ana lovri	bandić	invalidi	ivo sanac	rukomet	sessa	split	
10	5792	fašizam,	fašizam,	63	8	15	24	44	55	63	#VALUE!	#VALUE!	fašizam	istra	italija	labinska	mussolin	rudari		
11	6220	ministar	ministar	38	19	38	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	ministar	reforma z						
12	6606	kriminalc	kriminalc	34	11	23	34	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	kriminalc	političari	sindikati					
13	6635	državni p	državni p	37	17	23	37	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	državni p	hzmo	krizni nar					
14	6659	političke	političke	18	18	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	#VALUE!	političke							

Figure 3.3.
Sheet containing extracted keywords

In the end of the preprocessing phase we have constructed a keyword sheet (*tag-matrix*) that can be used in the network visualization tool we have implemented. The only difference to the previous sheet is that only the Article ID and actual keywords are shown and the Author field has been added, as shown on **Figure 3.4.**

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916



B2 =vlookup(A2;'articles-clean'!$B$2:$K$784;10)

	A	B	C	D	E	F	G	H	I	J
1	ArticleID	Author	Tag							
2	0	MKn	јосипови	азил	аутоцес	дубровн	миланов	пупавац	пусић	србија
3	5322	marival	ivo sanac							
4	5417	marival	dnevnik	eu	komenta	pollitika				
5	5463	marival	boris kun	dnevnik	gospodar	hdz	počela el	poslodav	sindikati	sinergija
6	5504	marival	bdp	ivo sanac	razvoj	sindikati	vlada	vođa		
7	5529	marival	građani	izbori	narod	sindikati	vlada	vođa		
8	5538	marival	ana lovrii	bandić	invalidi	ivo sanac	rukomet	sessa	split	
9	5792	marival	fašizam	istra	italija	labinska	mussolin	rudari		
10	6220	marival	ministar	reforma ;						
11	6606	marival	kriminalc	političari	sindikati					
12	6635	marival	državni p	hzmo	krizni nai					
13	6659	marival	političke							
14	6870	marival	krizni poi	samostal	vlada					
15	6872	sjenka	12 zvjezc	autokanil	globalna	hobotnica	izabrani	kleptoma	vojnici pl	
16	6914	sjenka	croatorur	djeva i bi	kleptoma	korporati	paralelna	puzajuća	sjeverni j	
17	6950	sjenka	12.i21.12	dan zaok	istina o č	novi poče	politika i			
18	6965	sjenka	hrvatska	v.stengl	zatvor					
19	6977	sjenka	hdz	kazaliste	sdp	tojanski l				
20	7003	sjenka	balkan	henry kis	ivo zanac	louis blou	poker	ringišpil		
21	7029	sjenka	financijsk	nafta	šuplji dol	zlatna po				
22	7052	sjenka	inplantat	manipula	matrix	obrazova	oslobođe	znanje		

Figure 3.4.
Final matrix of extracted keywords

In order to be used by subsequent tools, the preprocessed sheets have to be saved in CSV format, which is a standard option in most spreadsheet programs. In this way we have exported the *commenter-network*, *voter-network* and *tag-matrix* sheets to individual CSV documents.

4. Natural Language Processing

Natural Language Processing (NLP) represents a set of tools used in computational linguistics which make it possible to collect, prepare and analyze digital (textual) information which is accessible for example via WWW. The goal of NLP is to define possible ways for extracting tacit knowledge from such information. Research efforts for computation analysis of content on Croatian language are, although numerous, limited to several researchers. Not all of these tools are ready to be used out of the box. The majority of them require some kind of preparation for computational use. **Table 4.1** gives an outline of available linguistic tools for the Croatian language.

Table 4.1. List of available Croatian linguistic tools

ISSN 2285 – 4916
ISSN-L 2285 - 4916

Tool name	**Details available in:**
A simple rule-based stemmer for Croatian	http://nlp.ffzg.hr/resources/tools/stemmer-for-croatian/
Croatian National Corpus	(Tadić, 2009)
SETimes.HR corpus and dependency treebank of Croatian	(Agić, Ljubešić & Merkler, 2013) (Ljubešič, Stupar, Jurič, *et al.*, 2013)
hrWaC2.0 – Croatian web corpus	http://nlp.ffzg.hr/resources/corpora/hrwac/
hrenWaC – Croatian-English Parallel Web Corpus	http://nlp.ffzg.hr/resources/corpora/hrenwac/
SETimes - A Parallel Corpus of English and South-East European Languages	http://nlp.ffzg.hr/resources/corpora/setimes/
NER: Named entity recognition	(Ljubešič, Stupar, Jurič, *et al.*, 2013)
Syntactic dependency parsing	(Agić & Merkler, 2013)
Morphosyntactic tagging and lemmatization	(Agić, Ljubešić & Merkler, 2013)
Croatian WordNet	(Raffaelli, Tadić, Bekavac, *et al.*, 2008)
Morphological/Inflection/Lemmatization Engine for Croatian language	https://pypi.python.org/pypi/text-hr

When analyzing textual information in any language, although specific resources do exist, practical application that are ready to use for any interested user are very scarce. This is due to the fact that each possible analysis is very specific in set of resources used, order of their usage in use sequence and the expected results (be it in structure or in file

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

format). For the purposes of this article, and as a showcase of different possibilities when performing textual content analysis (in practice, in any language but in our study limited to Croatian language texts) we have developed a small, web based tool available for anyone. Although the application, called Croatian textual analyzer, works with any resource written on Croatian we have also supplied different sources that can be used in testing the application.

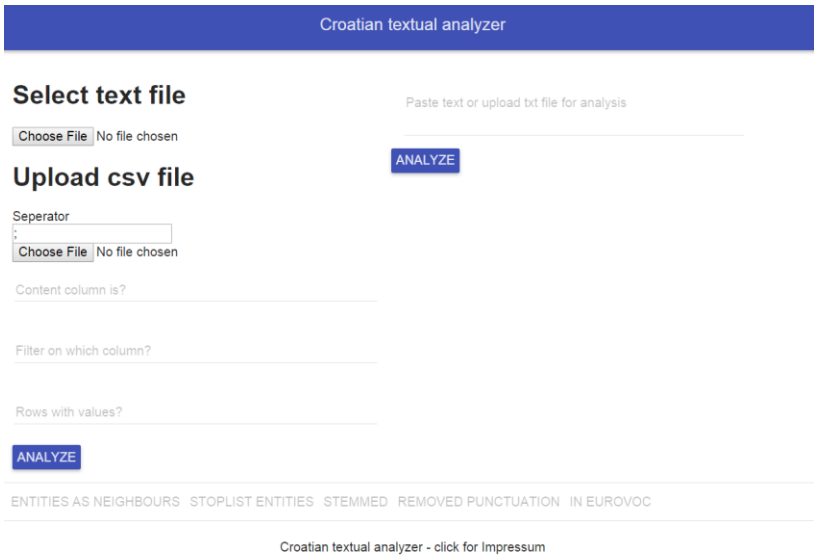


Figure 4.1.
Croatian textual analyzer - main screen

Figure 4.1 presents the interface developed for this tool. The interface is comprised of several main areas. The tools name, *Croatian textual analyzer*[7] (CTA), is stated at the top of the page. The main content is divided in two columns; left column is used to upload either a plain textual file whose content one wants to analyze (section *Select text file)* or a CSV file for analysis (*Upload CSV file;* when one wants to analyze multiple textual resources). The right column is used when one wants to analyze text which can be copied from the original source and passed it into the application. After the resource(s) to be analyzed are uploaded into the application, the analysis is started by pressing the *Analyze* button.

[7] Available at http://rec.foi.hr:5252/

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

When one wants to analyze a single document there are two ways to do so by using CTA. Firstly, a document or parts of it can be pasted into the input field, as shown in **Figure 4.2.** Additionally, one can store a single document in to locally stored plain text file. Once the file successfully uploads to our application, its content will be presented in the same area where we can paste the content when directly inserting content for analysis to the application.

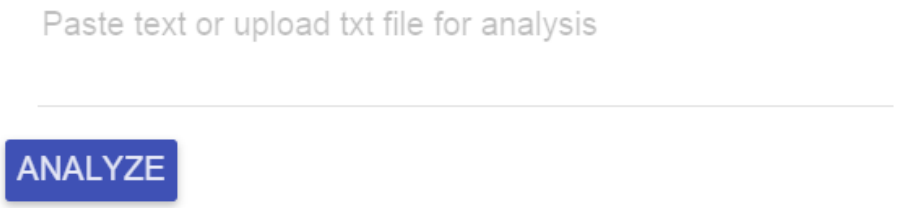


Figure 4.2.
Analyzing text by entering text into the input field

If we upload either a plain text or a CSV file for analysis the interface changes as shown in **Figure 4.3**. First, the name of the uploaded file is shown next *Choose file* button. Additionally, if a CSV file is uploaded the column in the file where the content one wishes to analyze is located has to be named. If there are specific rows one wants to analyze, depending on the value in whichever CSV file filed in the uploaded CSV file, one can state the name of that column and the values in that column that one wishes to filter out.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

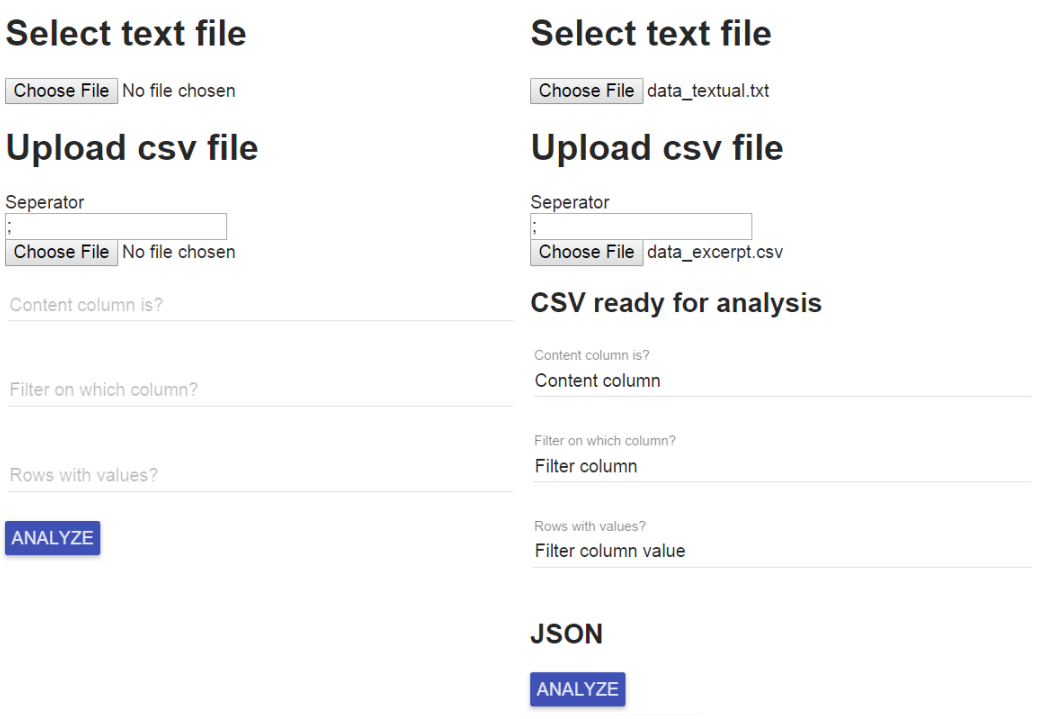


Figure 4.3.
File upload interface: before and after uploading

After the data is provided to the application, in either of the three possible ways described above, one can proceed to the actual analysis of the prepared text. The analysis, steps undertaken and returned results depend on the input. Once the results of the analysis are returned they can be downloaded. The difference between the menus for analysis results download is presented in **Figure 4.4**.

When the results are present the buttons for each type of the analysis are activated. If one decides to analyze a plain text file or textual content presented via entering text into the input field all buttons are activated, whilst if a CSV file is uploaded for analysis only *Entities as Neighbours* is available for download.

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

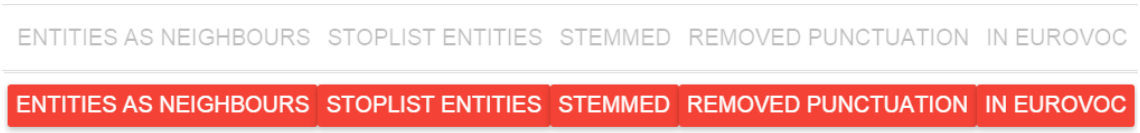


Figure 4.4.
Results analysis download interface: before and after analysis

For the purpose of this analysis we have prepared a set of resources, written in Croatian, which can be used to reproduce the results of this research effort. Those files are available for download on the CTA application by clicking on *Croatian textual analyzer - click for Impressum*. This view provides links to both input files, in both resources, as well as results achieved by our analyzer.

The tool processes input content in several steps. When dealing with a single document application prepares the content using three different tools, in the following order:

- *remove punctuation*
- *perform stemming*, based on the stemmer defined in (Ljubešić, Boras & Kubelka, 2007)
- *remove tokens that are not in Croatian addition to WordNet*, based on a list produced by our tool
- After the preparation, two entity search analysis are performed:
- *stopwords entity search*, based on entities in EuroVOC and entities that are either Croatian names or surnames.
- *Neighbor entity search*, where a specific grammar rule is used. The rule states that all entities are names with first capital letter.

The goal of the preparation steps is to reduce the number of tokens and remove recognized tokens which don't contribute to the meaning of the sentence and text in general. The effects of this reduction as well as the results for the two search analysis are shown in Table 4.2. Values represent the number of words in the document after each step.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Table 4.2. Reduction and analysis results

Analysis step	**Original size**	**Removing punctuation**	**Perform stemming**	**Apply stopwords**	***Stopwords entity search***	***Neighbor entity search***
Analysis results						
Words	*1006*	*1006*	*629*	*103*	*33*	*80*
Unique words	*-*	*586*	*407*	*54*	*17*	*64*

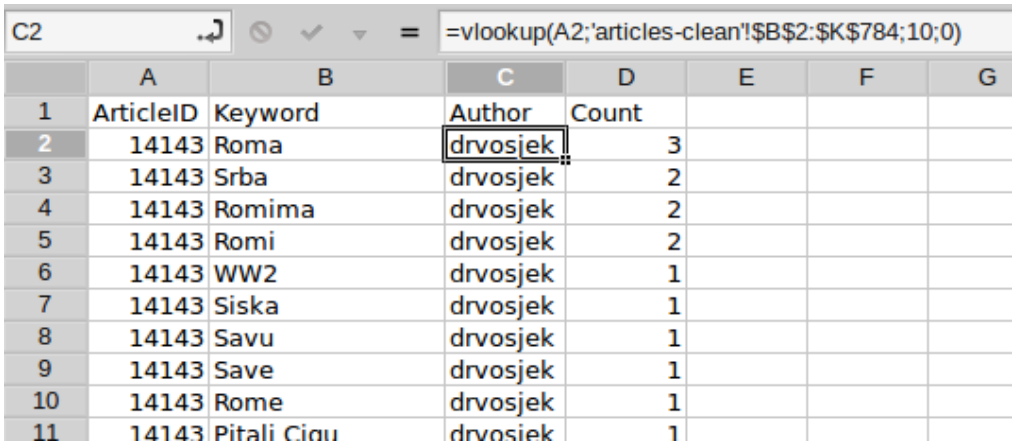
Based on the achieved results various further steps can be applied. In our case we visualized the data and thus linking individual document with the entities used in the document. The actual recognized concepts are shown in **Table 4.3**. When dealing with CSV file analysis, the system return only the results of *Entity search 2*.

Table 4.3. Frequency analysis results for entity search

Entity search 1	**Entity search 2**
('kad', 7), ('medij', 4), ('rad', 2), ('dan', 2), ('dug', 2), ('tih', 2), ('razlog', 2), ('dobr', 2), ('id', 2), ('dok', 1), ('vlast', 1), ('si', 1), ('francuz', 1), ('mah', 1), ('ljub', 1), ('prodan', 1), ('bank', 1)	*(u'Tito', 6), (u'Tita', 3), (u'Evropi', 3), (u'Kako', 2), (u'Hrvatskoj', 2), (u'3', 2), (u'Hrvata', 2), (u'4', 2), (u'35', 2), (u'Jesu', 2), (u'Titovog', 1), (u'5', 1), (u'Milijun', 1), (u'Staljina', 1), (u'KNOJ', 1), (u'Kolinde', 1), (u'Hitlera', 1), (u'Njema\u010dkoj', 1), (u'ETF', 1), (u'\u0106esi\u0107a Rojsa', 1), (u'80', 1), (u'138', 1), (u'D\u017eon Vejn', 1), (u'Tito 2', 1), (u'Dresden', 1), (u'Ovrhama', 1), (u'Teheranu', 1), (u'Njema\u010dko', 1), (u'Da', 1), (u'1', 1), (u'Englezi Amerikanci', 1), (u'Tu\u0111mana', 1), (u'Stra\u0161no', 1), (u'Evrope', 1), (u'Njema\u010dku', 1), (u'Englezima', 1), (u'500', 1), (u'A', 1), (u'Nagasaki', 1), (u'OZNA', 1), (u'Svakako', 1), (u'Ho\u0107e', 1), (u'Gospodarstvo', 1), (u'Vatikanom Od', 1), (u'Zagrebu', 1), (u'Znanost', 1), (u'Krle\u017eu', 1), (u'Demografija', 1), (u'Jalti', 1), (u'Crvene', 1), (u'Holivud', 1), (u'Ogla\u0161iva\u010di', 1), (u'Urbanizam', 1), (u'Hiro\u0161imu', 1), (u'Normandiji', 1), (u'Predsjednica', 1), (u'Paveli\u0107a', 1), (u'Delo\u017eacijama', 1), (u'Itd', 1), (u'Koliko', 1), (u'Skrene\u0161', 1), (u'Francuzi', 1), (u'Hrvatima', 1), (u'Sibir', 1)*

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

In order to be visualized the resulting CSV file has to be adjusted a bit. We added the resulting data to our spreadsheet in a new sheet (*nlp-matrix*) and added an Author column using a *VLOOKUP* formula as shown on **Figure 4.5**.


C2 =vlookup(A2;'articles-clean'!$B$2:$K$784;10;0)

	A	B	C	D	E	F	G
1	ArticleID	Keyword	Author	Count			
2	14143	Roma	drvosjek	3			
3	14143	Srba	drvosjek	2			
4	14143	Romima	drvosjek	2			
5	14143	Romi	drvosjek	2			
6	14143	WW2	drvosjek	1			
7	14143	Siska	drvosjek	1			
8	14143	Savu	drvosjek	1			
9	14143	Save	drvosjek	1			
10	14143	Rome	drvosjek	1			
11	14143	Pitali Ciqu	drvosjek	1			

Figure 4.5.
The *nlp-matrix* sheet

5. Social and Conceptual Network Visualization

Visualization of complex social, conceptual or other networks is an art and a science. Such visualizations have to be appealing, informative and concise. There are numerous algorithms for complex network visualization that have been proposed and developed, each of which adjusted to different kinds of networks. One of such is the k-core decomposition algorithm proposed by (Alvarez-Hamelin et al., 2005) which we will employ for our example study. Particularly, we will use the LaNet-vi[8] tool, for which we have implemented a web based interface that allows us to use our custom made CSV files from section 3.

The interface (shown on **Figure 5.1**) has a simplistic interface which allows the user to paste a CSV file into an input box, choose the appropriate file format, and visualize the content of the file.

[8] Available at http://lanet-vi.soic.indiana.edu/

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Figure 5.1.
Results analysis download interface: before and after analysis

There are five possible selections:

- Adjecency list (for visualizing the *commenter-network* and *voter-network* CSV files)
- Tripartite Hypergraph list -1st node (for visualizing the *tag-matrix* CSV file by article)
- Tripartite Hypergraph list -2nd node (for visualizing the *tag-matrix* CSV file by author)
- Weighted Hypergraph - 1st node (for visualizing the *nlp-matrix* CSV file by article)

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

- Weighted Hypergraph - 2st node (for visualizing the *nlp-matrix* CSV file by author)

Firstly, we have visualized the two social networks (*commenter-network* and *voter-network*) shown on **Figures 5.2** and **5.3**, respectively. The size of the node (shown on each left hand side of a visualization) corresponds to the degree of the node (e.g. the number of connections the node participates in), while the color of the node (shown on each visualizations right hand side) corresponds to the shell the node is part of (e.g. a set of nodes which are mutually well connected).

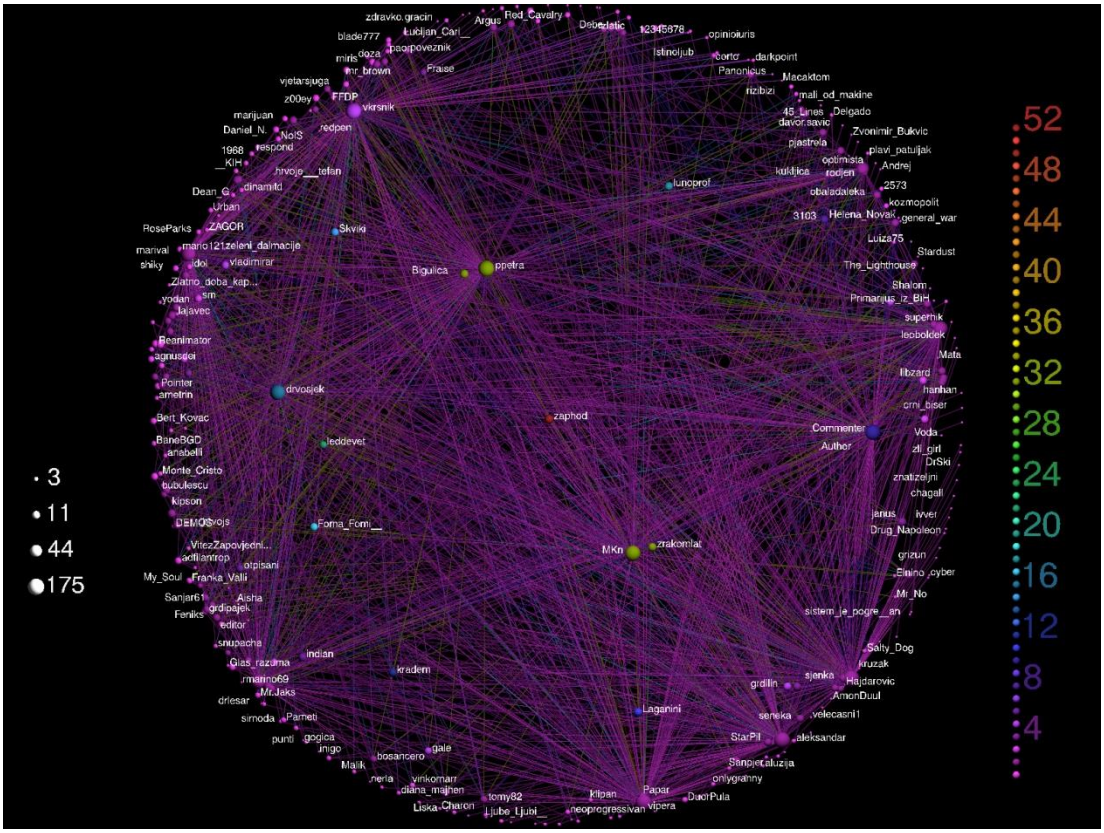


Figure 5.2.
The commenter social network

In the commenter social network one can observe a few central nodes, whilst all the other nodes are in the most outer shells. This is expected, since we have analyzed only 12

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

blog authors to which articles other users have commented. Thus, the authors are most central when compared to other users.

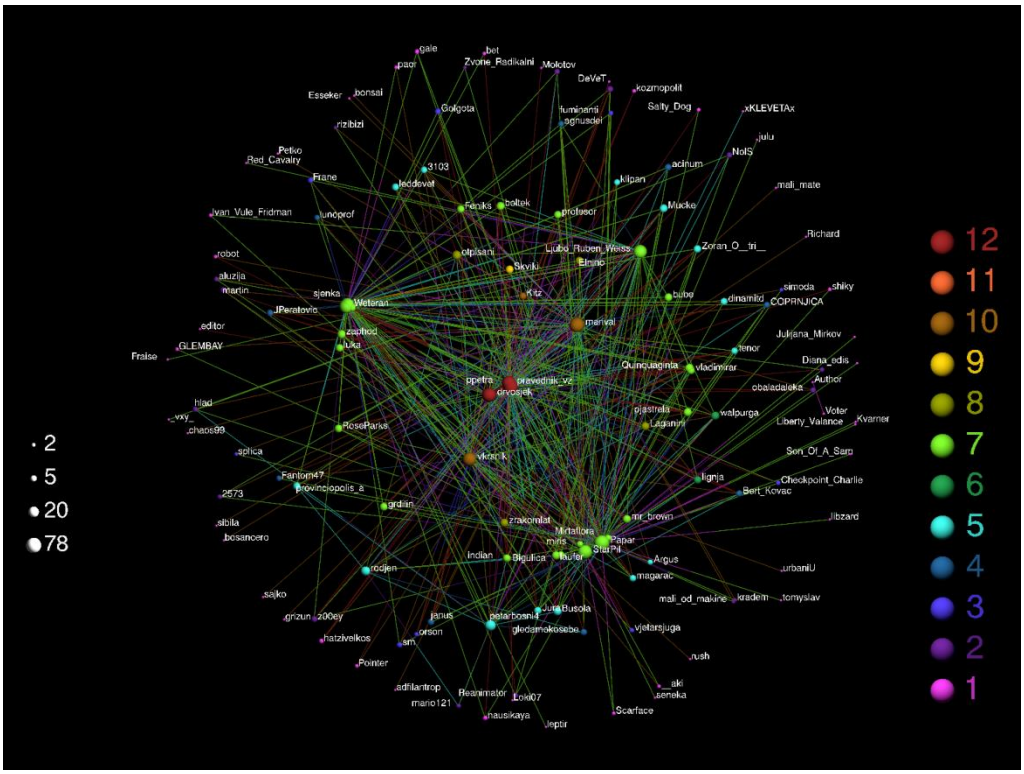


Figure 5.3.
The voters social network

In the voters social network, on the other hand, most central are bloggers that received most up-votes from other users, as well as users which gave the most votes to others. When comparing these to visualizations, one can observe a difference in network statistics: there are much more connections in the commenter network, meaning that there were more comments to articles than there were votes.

After visualizing the social networks, we have visualized the four conceptual networks: two acquired through mining user supplied tags and two acquired through natural language processing. Of each group one network was constructed based on a conceptual criterion (e.g. concepts were connected if bound to the same article) and one based on a social criterion (e.g. concepts were connected if used by the same author). The four network's visualizations are shown on **Figures 5.4, 5.5, 5.6** and **5.7**, respectively.

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

When comparing these four networks, one can observe that they are very different in a number of aspects. As a first, nodes in the article based networks have quite a smaller degree than the user based networks. This is due to the fact that users use more concepts on various articles, whilst articles usually have a limited number of keywords. User supplied tag based networks are better connected than NLP based networks. The reason for this is that author's supply tags based on their internal conceptual networks and usually use a relatively small number of concepts when tagging content. NLP on the other hand acquires concepts based on text statistics and semantics, and thus the number of concepts is greater. Additionally, this is due to the limitations of the used NLP tools, e.g. some concepts might have been connected if their morphological roots would have been analyzed as well.

When analyzing the actual content of the networks, we can observe that in the article tag network two most central concepts are *Deutsche Telekom* and *Ivica Mudrinić* (who is the CEO of T-Com in Croatia). It seems that a number of articles in the analyzed dataset have dealt with issues concerning the work of Deutsche Telekom in Croatia.

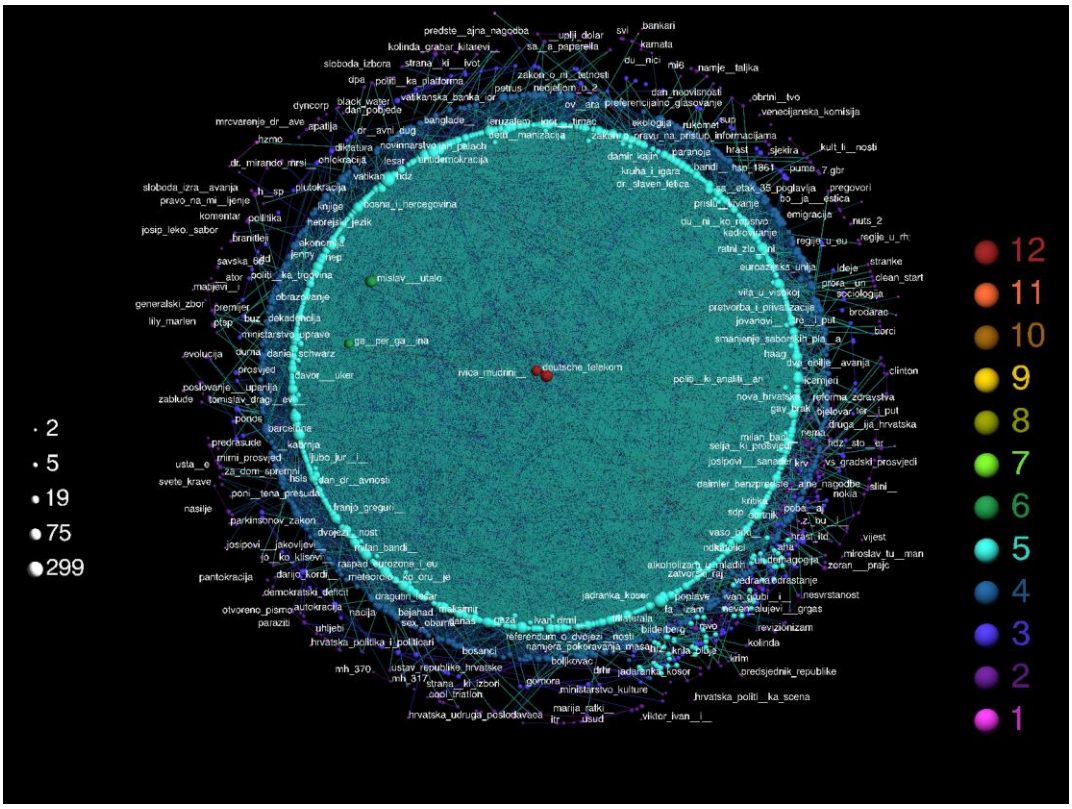


Figure 5.4.
Conceptual network of user supplied tags used on the same article

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

In the tag author network, the most central concepts are *MUP* (the Croatian ministry of internal affairs), *profesionalizacija* (professionalization), and *predsjednički izbori 2005* (presidential elections of 2005. From these results one might expect that the analyzed users have a great interest in these topics.

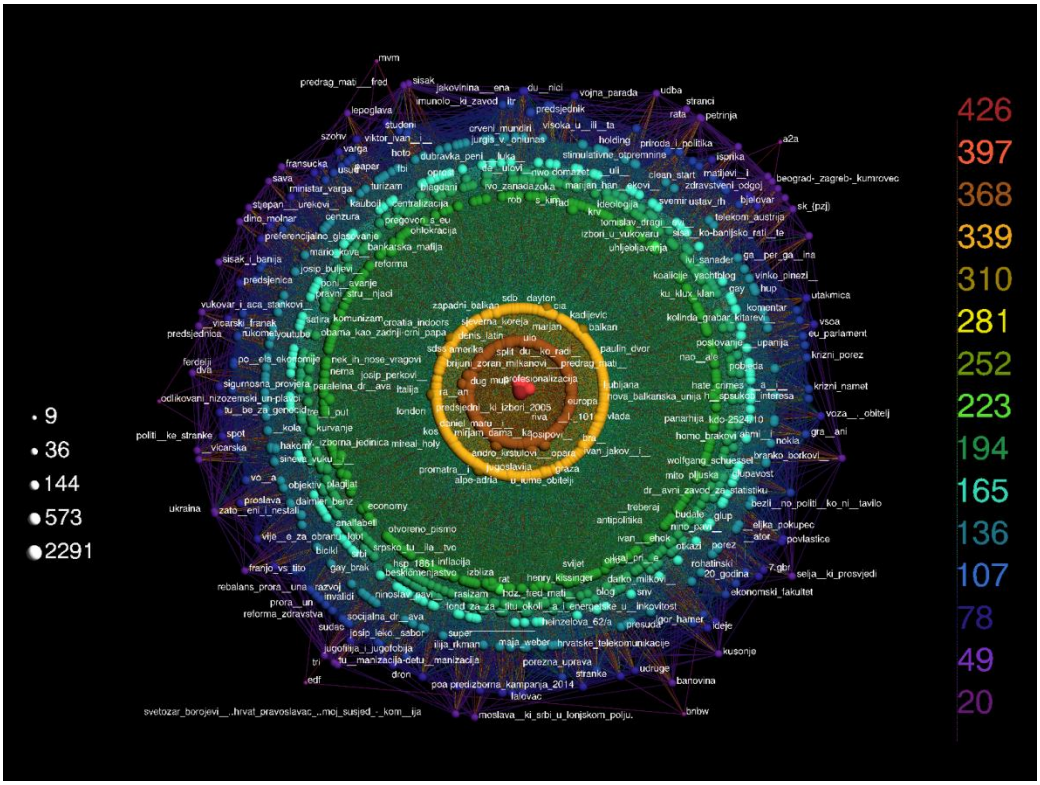


Figure 5.5.
Conceptual network of user supplied tags used by the same author

In the NLP article network, there are two central shells (one towards the middle of the visualization, and one towards the right). The middle shell does not seem to have an obvious central concept, but most central is the concept *Rimljani* (Romans). The right shell has a number of central concepts including *Savske 44* (an address in Zagreb close to a protest by the Croatian veterans which is ongoing from November 2014 until the time of writing this article), *Feral Tribune* (a satirical political newspaper which was popular in Croatia during the 1990-ies recently often cited in the context of Charlie Hedbo), *SDPovci* (an informal name for the currently ruling social democratic party members), *SDPom*

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

(another informal name for the same party), *Klemma* (Josip Klemm one of the informal organizers of the previously veteran mentioned protest).

While the concepts in the latter shall make sense for a political blogging site, the term dealing with the Romans required us to review the actual articles where the concept has been used. It is due to the fact that Croatian people like to cite latin sayings which is then attributed to the Romans.

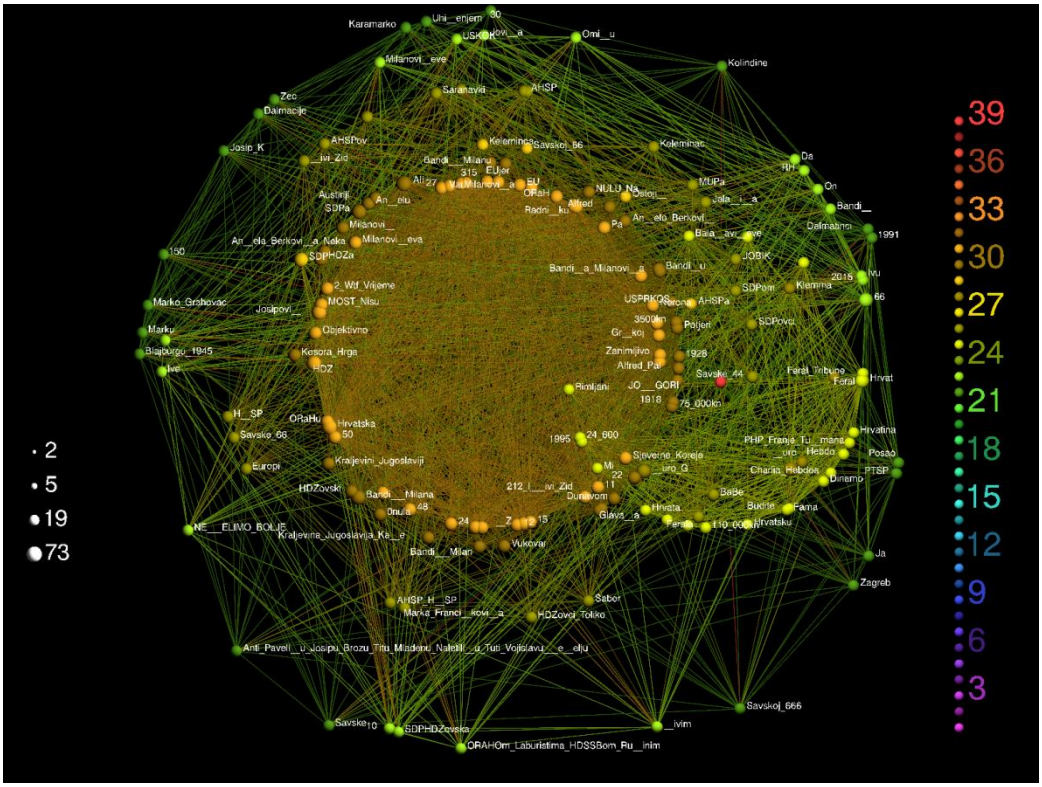


Figure 5.6.
Conceptual network of keywords extracted from the same article

The NLP author network also has two shells. The right one doesn't have any central concepts, while the middle one has only one *Hrvatskoj* (Croatia). The right shell consists in full of keywords written in Cyrillic letters (due to constraints of LaNet-vi, non-ASCII characters cannot be visualized) which is due to the fact that one of the analyzed bloggers has used Cyrillic letters on one of his articles, but they weren't used anywhere else, which is why these keywords didn't connect to any other concepts.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

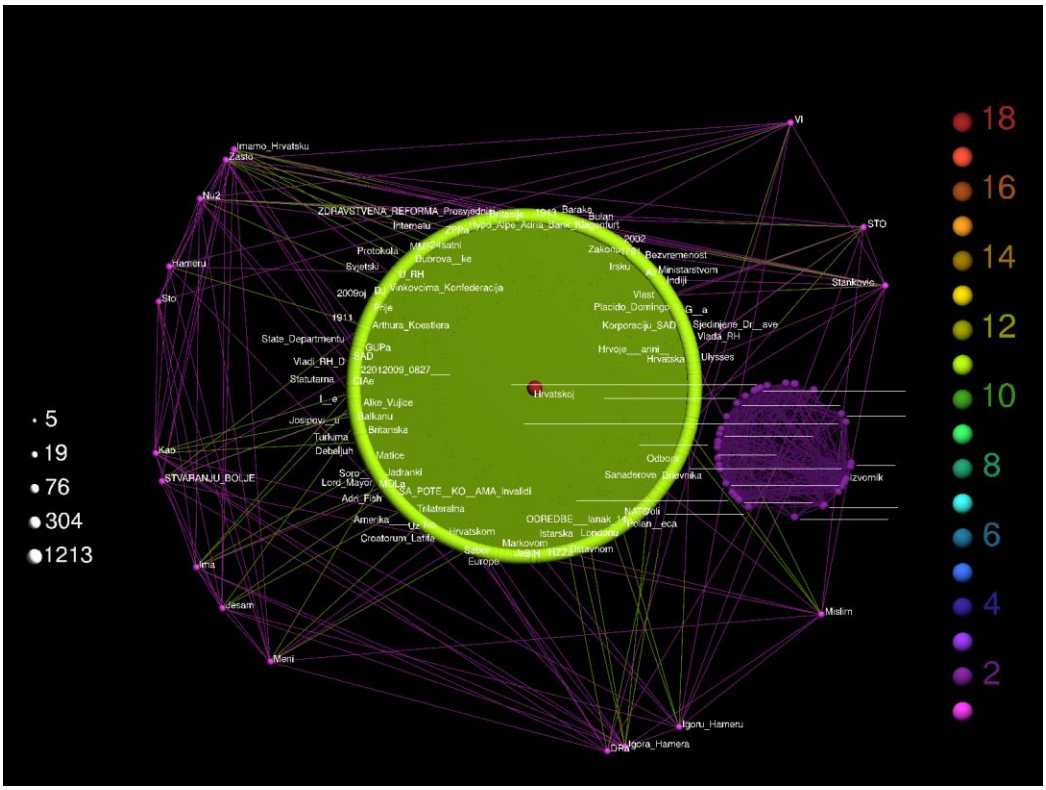


Figure 5.7.
Conceptual network of extracted keywords used by the same user

As from these four visualizations of conceptual networks, we can conclude that the NLP article network provides us with the best insights into the social discourse since it shows two partially connected shells of concepts that reflect the current political situation in Croatia (current protest, the current government, current discourse on world issues). Since the study was limited on only 12 blog authors, a more detailed study taking into account more authors or even better all of them by using the same toolset but concentrating only on NLP article networks would be advisable.

6. Conclusions

In this chapter we have covered the details and the possibilities of analyzing unstructured, textual information available over the WWW. This work has a goal of introducing the reader with the possibilities and tool available for extracting tacit

ISSN 2285 – 4916
ISSN-L 2285 - 4916

knowledge from such sources. The entire process has been broken down into four parts, each dealing with a specific step in the analysis. Section 2 analyzed how we can collect information from online resources like blogs, news portals or any other digitally available address. These resources contain valuable information about the text, opinions and entities that are mentioned in the text and users' attitudes towards the content through their comments and given ratings. To demonstrate the process of accessing such content we have provided an example and shown the interface and possibilities of RapidMiner. RapidMiner is a powerful tool: once the data is collected and prepared, which gives several other possibilities for data analysis beside web content crawling. Before the collected data can be analyzed, we have to prepare it. This process depends on the type of information collected as well as the type of analysis applied to the data. Section 3 gives the specifics of the process of data preparation for our chapter. We use the CSV file format which is widely used as it is supported by a large number of tools used in data analysis. Additionally, CSV files give us the opportunity to structure collected data with help of delimiters. This chapter is focused on the analysis of written data accessible over the web therefore, section 4 focuses on the possibilities we have in accessing and analyzing a large amount of textual information. For the purposes of this analysis we have developed a web application, called *Croatian textual analyzer,* which can be used for entity recognition and frequency analysis of textual information. The application is based on the tools available for NLP of content written in Croatian. The actual tools used vary between languages. Unfortunately, not all European languages have the same level of tools for such a task. English is still the most supported language and the number and the quality of NLP tools for that language is vast. The rest of tools have limited resources that can be used in textual content analysis in each respected language. The developed application, its interface, the tools used and the results it returns have also been described. The results of this tool are then used in the final step of our research: we are focusing on social and conceptual network visualization of collected, pre-processed and prepared data. For this sake we have implemented an additional tool to ease the visualization of the prepared data. This process and the results are explained in section 5 of this chapter. Our tutorial focuses on analyzing connections between the content of the article, authors and on-line users who interact with the article. By visualizing prepared information, as described in previous sections, concepts that are most important are visually highlighted. Several different networks are presented and explained.

The final results of the example study indicate that the conceptual network based on keywords extracted by the implemented NLP tool and constructed by through the rule that two keywords are connected if they were extracted from the same article, shows the best insight into the current political and social discourse in Croatia. The results also indicate that the used dataset based on only 12 blog authors is too small to provide a better

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

understanding of the internal conceptual structure of this discourse, which is why in future studies a greater sample has to be used. Another interesting study would be to analyze the processes that build up the constructed networks through a temporal perspective. These and similar considerations are subject to our future studies.

European Quarterly of Political Attitudes and Mentalities EQPAM
Volume 4, No.3, July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Appendix

Tools, Datasets, Results and Other Resources

In this work we have used several tools that are freely available for use to interested readers. Besides open source tools, this work uses two web applications specially developed for this chapter. Here you can find a list of resources used and their respective on-line addresses where they are available.

Tools:

1. RapidMiner, available at https://rapidminer.com/
2. Gnumeric, available at http://www.gnumeric.org/
3. LibreOffice Calc, available at http://www.libreoffice.org/
4. Croatian textual analyzer, available at http://rec.foi.hr:5252/
5. Large network visualizer, available at http://ai.foi.hr/netviz
6. LaNet-Vi, available at http://lanet-vi.soic.indiana.edu/

Datasets:

1. RapidMiner process set up for this chapter is available at http://ai.foi.hr/netviz/download/web-mining.rmp
2. Links used as input to the RapidMiner process are available at http://ai.foi.hr/netviz/download/web-mining-links.csv
3. Entire dataset collected from Pollitika.com, available at http://ai.foi.hr/netviz/download/data_original.csv
4. Preprocessed dataset, available at
   http://ai.foi.hr/netviz/download/dataset.gnumeric
   http://ai.foi.hr/netviz/download/dataset.ods
   http://ai.foi.hr/netviz/download/dataset.xls
5. Sample dataset, generated from 3., available at http://ai.foi.hr/netviz/download/data_excerpt.csv
6. A single article, extracted from 3., available at http://ai.foi.hr/netviz/download/data_textual.txt

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Results:

1. NLP CSV file results, available at http://ai.foi.hr/netviz/download/CSVResuls_POS_Neighbours.csv
2. NLP TXT file results: entities in EuroVOC, available at http://ai.foi.hr/netviz/download/TXTResults_In_EuroVOC.csv
3. NLP TXT file results: advanced entities analysis, available at http://ai.foi.hr/netviz/download/TXTResults_POS_Neighbours.csv
4. NLP TXT file results: removed stop words, available at http://ai.foi.hr/netviz/download/TXTResults_Removed_grammar.csv
5. NLP TXT file results: stemming, available at http://ai.foi.hr/netviz/download/TXTResults_Stemmed.csv
6. NLP TXT file results, available at http://ai.foi.hr/netviz/download/TXTResults.csv
7. Commenter social network visualization, available at http://ai.foi.hr/netviz/download/commenter-network.png
8. Voter social network visualization, available at http://ai.foi.hr/netviz/download/vote-network.png
9. Tag article conceptual network visualization, available at http://ai.foi.hr/netviz/download/tags-articles.png
10. Tag author conceptual network visualization, available at http://ai.foi.hr/netviz/download/tags-authors.png
11. NLP article conceptual network visualization, available at http://ai.foi.hr/netviz/download/nlp-articles.png
12. NLP author conceptual network visualization, available at http://ai.foi.hr/netviz/download/nlp-authors.png

Other resources:

1. https://greenido.wordpress.com/2013/12/16/big-query-and-google-spreadsheet-intergration/
2. https://netlytic.org/home/?p=168
3. http://vancouverdata.blogspot.in/2010/11/text-analytics-with-rapidminer-loading.html
4. http://vancouverdata.blogspot.in/2011/02/how-to-web-scraping-xpath-html-google.html
5. http://www.w3schools.com/xpath/
6. http://www.regular-expressions.info/tutorial.html

ISSN 2285 – 4916
ISSN-L 2285 - 4916

Acknowledgements

This work has been supported in part by the Croatian Science Foundation under the project number 8537.

References

Agić, Ž., Ljubešić, N. & Merkler, D. (2013) Lemmatization and Morphosyntactic Tagging of Croatian and Serbian. *Proceedings of the 4th Biennial International Workshop on Balto-Slavic Natural Language Processing*. 48–57.

Agić, Ž. & Merkler, D. (2013) Three Syntactic Formalisms for Data-Driven Dependency Parsing of Croatian. In: Ivan Habernal & Václav Matoušek (eds.). *Text, Speech and Dialogue*. pp. 560–567.

Alvarez-Hamelin, J. I., Dall'Asta, L., Barrat, A., & Vespignani, A. (2005). Large scale networks fingerprinting and visualization using the k-core decomposition. In *Advances in neural information processing systems*. pp. 41-50.

CenturyLink (2011) Big Data - Defining the Digital Deluge - Infographic, Available at <http://www.centurylink.com/business/artifacts/pdf/resources/big-data-defining-the-digital-deluge.pdf>, accessed: 20.04.2015.

Chomsky, N. (2002) *Syntactic structures*. 2nd ed. Walter de Gruyter.

Grisham, R. (2010) Information Extraction. In: A Clark, C Fox, & S Lappin (eds.). *The Handbook of Computational Linguistics and Natural Language Processing (Blackwell Handbooks in Linguistics)*. Blackwell Publishing Ltd. pp. 517–530.

Jackson, P. & Moullinier, I. (2002) *Natural Language Processing for Online Applications*. Ruslan Mitkov (ed.). John Benjamins Publishing Co.

Kao, A. & Poteet, S.R. (2006) *Natural language processing and text mining*. Anne Kao & Stephen R. Poteet (eds.). Springer Verlag.

Kapetanović, A., Tadić, M. & Brozović-Rončević, D. (2012) THE CROATIAN LANGUAGE IN THE DIGITAL AGE / HRVATSKI JEZIK U DIGITALNOM DOBU. In: *White Paper Series*. Springer.

Liu, B. (2011) *Web Data Mining: Exploring Hyperlinks, Contents and Usage Data*. 2nd ed. Berlin, Heidelberg, Springer Berlin Heidelberg.

Luhmann, N. (1984). Soziale systeme. Frankfurt am Main: Suhrkamp.

Ljubešič, N., Stupar, M., Jurič, T. & Agić, Ž. (2013) Combining Avalilable Datasets for Building Named Entity Recognition Models of Croatian and Slovene. *Slovenščina 2.0*. (2), 35–57.

Ljubešić, N., Boras, D. & Kubelka, O. (2007) Retrieving Information in Croatian : building a simple and effcient rule-based stemmer Sanja Seljan & Hrvoje Stančić (eds.). *Digital Information and Heritage*. 313–320.

Mander, J. (2015) Daily time spent on social networks rises to 1.72 hours, GlobalWebIndex, Available at <https://www.globalwebindex.net/blog/daily-time-spent-on-social-networks-rises-to-1-72-hours>, accessed: 20.04.2015.

Mika, P. (2007) Ontologies are us: A unified model of social networks and semantics. Web Semantics: Science, Services and Agents on the World Wide Web. 5(1), pp. 5–15.

Neumann, M., Srbljinović, A., Schatten, M. (2014) "Trust me, I know what I'm doing!" Competence Fields as a Means of Establishing Political Leadership. European Quarterly of Political Attitudes and Mentalities. 3(2), pp. 18-33.

Ogrodniczuk, M., Garabík, R., Koeva, S., Krstev, C., et al. (2012) Central and South-European language resources in META-SHARE. *INFOtheca - Journal of Informatics & Librarianship*. 13 (1), 3–26.

EQPAM Volume 4 No.3 July 2015
ISSN 2285 – 4916
ISSN-L 2285 - 4916

Pejić Bach, M., Schatten, M., Marušić, Z. (2013) Data Mining Applications in Tourism: A Keyword Analysis, In Hunjak, T., Lovrenčić, S., Tomičić, I. (Eds.) Proceedings of the 24th Central European Conference on Information and Intelligent Systems, Varaždin : Faculty of Organization and Informatics, pp. 26-32.

Raffaelli, I., Tadić, M., Bekavac, B. & Agić, Ž. (2008) Building Croatian WordNet. *Proceedings of the 4th Global WordNet Conference*. 349–359.

Roe, C. (2012) The Growth of Unstructured Data: What To Do with All Those Zettabytes?, DataVersity, Available at <http://www.dataversity.net/the-growth-of-unstructured-data-what-are-we-going-to-do-with-all-those-zettabytes/>, Accessed: 20.04.2015.

Schatten, M. (2011) Mining Social and Semantic Network Data on the Web, Seminar za metodologijo in informatiko, Novo Mesto, Faculty of Information Studies.

Schatten, M. (2013) What do Croatian Scientist Write about? A Social and Conceptual Network Analysis of the Croatian Scientific Bibliography. Interdisciplinary description of complex systems. 11(2), pp. 190-208.

Schatten, M., Rasonja, J., Halusek, P., Jakelić, F. (2011) An Analysis of the Social and Conceptual Networks of CECIIS 2005 – 2010, In Hunjak, T., Lovrenčić, S., Tomičić, I. (Eds.) Proceedings of the 22nd Central European Conference on Information and Intelligent Systems, Varaždin : Faculty of Organization and Informatics, 2011. pp. 259-264.

Schatten, M., Ševa, J., Okreša Đurić, B. (2015) An Introduction to Social Semantic Web Mining & Big Data Analytics for Political Attitudes and Mentalities Research. European Quarterly of Political Attitudes and Mentalities EQPAM. 4(1) pp. 40-62.

Statista (2014) Statistics and facts about Social Networks, Available at <http://www.statista.com/topics/1164/social-networks/>, accesed: 20.04.2015.

Tadić, M. (2009) New version of the Croatian National Corpus Dana Hlaváčková, Aleš Horák, Klara Osolsobě, & Pavel Rychlý (eds.). *After Half a Century of Slavonic Natural Language Processing*. 199–205.

Voinea, C. F., Schatten, M. (2015) Recovering the Past. Eastern European Web Mining Platforms for Reconstructing Political Attitudes. European Quarterly of Political Attitudes and Mentalities EQPAM. 4(1), pp. 22-39.