DOM-based Keyword Extraction from Web Pages

Himat Shah
School of Computing
University of Eastern Finland
Joensuu Finland
himat@cs.uef.fi

Mohammad Rezaei
School of Computing
University of Eastern Finland
Joensuu Finland
rezaei@cs.uef.fi

Pasi Fränti *
School of Computing
University of Eastern Finland
Joensuu Finland
franti@cs.uef.fi

ABSTRACT

We present D-rank, an unsupervised, language and domain independent method for automatically extracting keywords from a single web page. The method does not use any corpus, and relies only on the information and features on the web page including page URL, word frequency, title, hyperlinks, and headers, which are extracted from DOM tree of the page. Different scores are assigned to the words according to their importance that is specified by their positions in the web page. Experimental results on web pages in three different languages show the effectiveness of the proposed method.

CCS CONCEPTS

• Information Systems • Information Retrieval • Retrieval tasks and goals • Summarization

KEYWORDS

Keyword extraction, Web page, DOM structure, Language independent, Unsupervised

1 Introduction

Today, there are more than 1.5 billion web pages on the internet. Finding the relevant web pages that the user is seeking is often a difficult task. In this regard, representative keywords or keyphrases assigned to each web page is very helpful. Keywords are useful for many purposes such as indexing, labelling, summarizing, highlighting, browsing, searching [1], clustering [2], and content-targeted advertising [3, 4]. Keyword extraction is the task of automatic identification of a set of words or phrases that best describe the web page [5].

There are numerous papers in the literature on the keyword extraction from text documents, which are often structured and following a standard writing formats and rules. On the other hand, only a few research papers have been published about extracting keywords from web pages, which are unstructured and heterogeneous in nature. Web pages often include irrelevant text that may make keyword extraction more difficult, such as advertisements, script codes, formatting and styling text, hyperlinks, and tags [3, 11]. Some of these side information texts, however, can be used to improve the accuracy of keyword extraction. Another difficulty with web pages is the variety of languages on the internet; sometimes, even one page with multiple languages is possible.

In this work, we aim at extracting keywords rather than key phrases. Extracting individual words might have certain advantages. Suppose that “amusement park” appeared only once in the text, but both “amusement” and “park” appeared many times separately. This approach would make it easy to use this knowledge [4].

Majority of existing keyword extraction methods use *natural language processing* (*NLP*) components including *stemming*, *part of speech* (*POS*) *tagging*, and *lemmatization*, which are language dependent, and makes it difficult to generalize a method for different languages. Our goal in this research is to extract only language-independent features from web pages so that our method would work with different languages.

Many different keyword extraction methods, both *supervised* and *unsupervised*, can be found in the literature [4, 9, 10, 13, 16, 17, 18]. In a supervised approach, a set of features is extracted from documents human labelled with keywords to learn a model [12, 18]. The results of this approach depends on the trained model and the dataset it is trained on. It does not perform well on documents from different domains/languages or style than the ones used to train the model [24]. Regarding web pages with diverse and emerging topics and writing styles, the supervised approach trained on a limited set of training datasets is unlikely to provide good results [8]. Furthermore, manual annotation of training data is time consuming and the quality of annotations can still be questionable since people can have different opinions about the suitable keywords for a web page [6]. In this paper, we therefore propose an unsupervised method to avoid the aforementioned limitations.

GenEx is the first supervised method in which a genetic algorithm learns to identify the best representative keyphrases from a list of candidate phrases. The candidate phrases are selected from all possible phrases in the text according to a set of heuristic rules [19]. KEA is a different supervised method, where the candidate keyphrases are first selected, and then, a Naïve Bayes classifier is trained given only two input features of a phrase: (tf-idf) term frequency-inverse document frequency and

AIIPCC 2019, December 19–21, 2019, Sanya, China

ACM ISBN 978-1-4503-7633-4/19/12$15.00
https://doi.org/10.1145/3331453.3331454

the distance from the beginning of the text [20]. Yih et al. [4] introduce a keyphrase extraction system for web pages, where a variety of features are extracted and given to logistic regression as the learning algorithm. The features are constructed from tf-idf, html Meta data and query logs.

Unsupervised approaches consider the keyword extraction task as a ranking problem [18]. Various methods have been proposed that can be classified into graph-based, clustering-based, and language modelling [11]. TextRank [5] is a state-of-the-art graph-based method, which builds a graph of the units (keyword or sentence) of the given text. The graph and the method for assigning a score to each node are inspired by Google PageRank [21]. Two nodes of the graph are connected if they co-occur in a window of maximum N lexical words (N=2 to 10). The candidate keywords are restricted using syntactic filters (POS tagging).

DegExt [15] is a graph-based, language and domain independent method. A graph is built on simple syntactic graph representations and considering the structural features of the document. A clustering based approach is used in [22] to guarantee that the document is covered both statistically and semantically by the keywords.

Another clustering based method is *Cl-rank* [11] in which nouns are selected as candidate keywords after POS tagging and lemmatization. They are clustered using agglomerative approach with *semantic similarity* measure. The importance of resulting clusters is determined by checking the distribution of its words among the web page. If the words of a *cluster* are related to only one text node of the web page, the cluster is considered unimportant. This technique ensures that the words from advertisements and other irrelevant text parts of the web page are not selected as keywords. To improve Cl-rank, *H-rank* [7] allows adjectives and verbs to be selected as keywords in addition to nouns. The reason for including adjectives and verbs is based on an experimental analysis on four different datasets, which shows that human labelled keywords for the web pages include 83% nouns, 13% adjectives, and 3% verbs.

This paper presents a new keyword extraction method called D-rank. It is for a single web page based only on the text and structural information of the page. In addition to term frequency, the *uniform resource locator* (*URL*) and several places of the web page including the *title*, *headings*, and *hyperlinks* provide valuable information about the important words in the web page, see Figure 1. We assign different scores according to the position of the words, and their frequencies. Top 20 highest ranked words are selected and then using simple search in *Wikipedia*, 10 most common words out of 20 are eliminated to achieve 10 keywords for the web page. The method is unsupervised, language and domain independent.

2 D-rank Keyword Extraction

Given a web page including its address or uniform resource locator (URL) and its content as a HTML file, we first pre-process the HTML content and URL to extract candidate keywords. We then give different scores to the words based on the information from HTML tags, which specify the positions of words. Our method, called D-rank, finally selects top 10 candidates as representative keywords for the web page. In this section, we first provide information on the importance of different positions of words in a web page and URL, and then, we explain how we use the positions to extract candidate keywords and score them. Figure 2 shows the features used in D-rank.

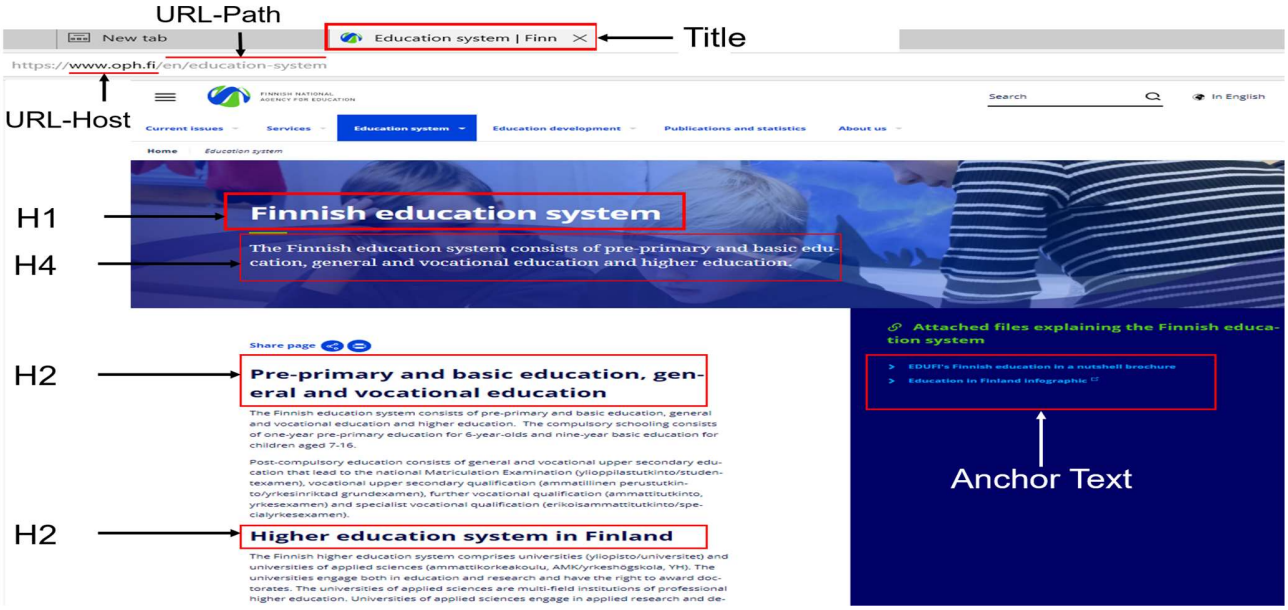

Figure 1: Title, URL and the Structure of an Example Web Page.

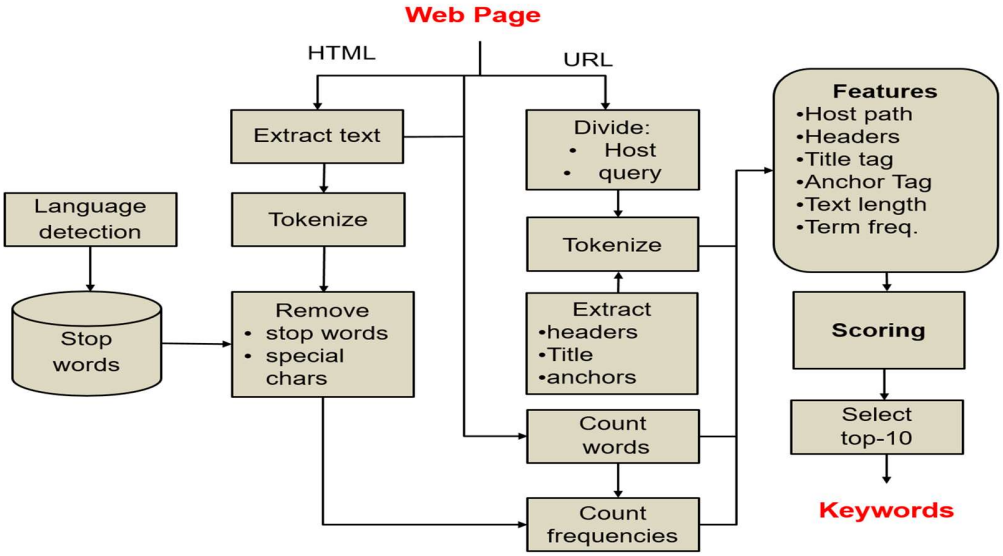

Figure 2: Workflow of the Proposed D-rank.

2.1 Word Position and Frequency

We score the words based on the places they appear in a web page: URL, title, six levels of headers, and hyperlinks, which provide important information to keyword extraction. In our scoring system, we also consider the frequencies of words.

URL is short but quite informative and content bearing. It usually includes important words related to the web page. There are web page classification methods, which use only the URLs of web pages. The important parts of a URL for keyword extraction includes *host*, *path*, and *query*, where the host and the path usually give more information than the query [23]. For example, *http://www.uef.fi/web/machine-learning/software* includes important keywords UEF (University of Eastern Finland) in the host part, and *machine learning* and *software* in the path part.

Header tags specify headings from level *h1* to *h6*, which structure the text in a web page. The headers often include important keywords of their following text. Therefore, we give a high score to the words in the headings. The scores decrease from the most important h1, to the least important h6.

The title tag and the Meta tag with *name=title* are important for finding the title of a web page, which provide the concise description of the web page. Search engines primarily find the topic of a web page by the text inside the title tag. The text appears at the top of a web browser when it shows the web page. Since they usually include important keywords, we give high scores for the words in the title.

Anchor tag, which defines a hyperlink, is another important source for keyword extraction. It has been widely used in automatic title extraction [14, 25]. We use the text of the anchor for our purpose, for example, the word BBC in <a HREF= ‘‘http://www.bbc.- com/’’>BBC. In [25], the authors use title and anchor text for describing and searching the topic from web documents.

Term frequency, counts the number of times that a word appears in a text. It indicates how important a word is. Almost all keyword extraction methods use the term frequency as a feature. A more frequent word should have more chance to be a keyword, and therefore, get a higher score. Of course, very common words such as stop words are exceptions and should be removed from the list of candidate keywords. One well-known solution for removing common words is using *tf-idf* instead of *tf* alone. However, in this paper, we aim at extracting keywords from a single web page, without relying on training documents or external sources.

2.2 Keyword Extraction Process

Our method includes following tasks in order to find candidate keywords: extracting actual text from HTML content, *cleaning* the text from symbols, *tokenizing* the text to have individual words, *detecting text language* and retrieving the list of stop words for the language, and removing stop words from the list of words. The same functions are performed on the URL to extract non-stop words. The words that appear in the page title, different levels of headers, and hyperlinks are classified in different lists. The lists are used in the scoring procedure to apply different scores to the words from different parts of the web page.

The process starts by building *document object model* (*DOM*) tree of the web page by parsing HTML tags. DOM tree provides complete structure and presentation of the HTML document. Text nodes of the tree are extracted to be used in keyword extraction. We *filter* and clean the text by excluding scripts, HTML structure symbols such as navigation lists, and style tags. Punctuation marks and special characters and symbols are also removed in this step. Pre-processing of the URL is also performed to extract three parts including host, path, and query.

In the next step, we tokenize the resulting text segments to provide individual tokens or words. Our method tokenizes the

URL and removes the symbols and special characters at the same time. We then remove the stop words from the list of words, which is an essential step in keyword extraction methods. Stop words are frequent words in different languages, which have least semantic value and should not be selected as keywords. We first detect the language of the text and then retrieve the list of stop words for the language. This does not contradict with our goal of providing a language-independent method because language detection tool and the list of stop words for different languages are widely available in small software libraries.

Usually the list of stop words for a specific language contains only a small number of common words. However, the number of common words, which should not be considered as keywords, are much more. To cope with this problem, we use a simple search in Wikipedia to determine the term frequency of words. When the goal is to extract k keywords, we initially select $2k$ high ranked words in the previous steps, and eliminate keywords according to the term frequency of the words in Wikipedia. We select k words out of $2k$, which have the fewest frequencies. We also calculate the frequency of each word in the entire text, and the total length of the text. The extracted information is summarized as follows:

- List of words and their frequencies
- List of words appeared in header 1, and similar lists for headers 2 to 6
- List of words in the page title
- List of words in the anchors texts (hyperlinks)
- List of words in the URL's host, and list of words in path and query together
- Number of all words in the page (N)

According to our experimental analysis, we introduce the following scoring method for a word with frequency f:

$$Score = 0.2 \times f, if\ N > 50 \quad (1)$$
$$Score = 0.5 \times f\ , if\ N \leq 50$$

Table 1 presents the scoring values for the words located in special places in the web page.

Table 1: Score for the Words in Special Positions of the Web.

Position of word	Score
H1	6
H2	5
H3	4
H4, H5, H6	2
Title	5
URL host	5
URL path and query	4
Hyperlink (anchor)	2

A candidate keyword can appear in several places, and therefore, its final score is calculated by combining the scores from different places. For example, when $N > 50$, if a word appears 6 times in the web page and it appears in h2, and the title, the score is calculated as follows:

$$\text{Score} = 6 \times 0.2 + 5 + 5 = 11.2$$

We select 20 words with highest scores, and eliminate 10 of them based on term frequency in Wikipedia, to finally have 10 words as representative keywords for the web page.

3 Experiments

We next evaluate the performance of the proposed keyword extraction method (D-rank) on four publically available datasets. We compare D-rank with three methods including Cl-rank [11], H-rank [7], and term frequency [26].

We use four datasets: *MACWorld* (English), *Guardian* (English), *Mopsi Services* (Finnish), and a *German* dataset. MACWorld dataset includes 220 web pages from Macworld.com. Each web page is an article about computer hardware, software, and technology with 5 to 10 assigned keywords given by articles' editors. Guardian dataset includes 421 news web pages from Guardian.com about different topics, with 5 to 15 keywords assigned to each page. Mopsi services from *http://cs.uef.com/mopsi* includes 414 web pages of location-based services in Finland. Different number of keywords from 1 to 10 has been assigned to different web pages. We also collected 100 German web pages on different topics. The ground truth keywords from 5 to 20 are taken from the web page metatag for keywords. The evaluation of the keyword extraction methods is based on three measures: precision, recall and F-score, which are generally used in evaluating information extraction systems. These measures are calculated based on true positive, false negative, and false positive, which are defined as follows:

True Positive = Number of correctly detected keywords.
False Positive = Number of incorrectly detected keywords.
False Negative = Number of keywords missed.

$$Precision = \frac{TP}{TP + FN} \quad (2)$$

$$Recall = \frac{TP}{TP + FP} \quad (3)$$

$$F\text{-}score = 2 \times \frac{precision \times recall}{precision + recall} \quad (4)$$

In comparing extracted keywords with the ground truth, usually the exact match between words is considered. We call this *hard evaluation*. However, this comparison does not reflect a good similarity between words. For example, if the ground truth keyword is *hotel* and the extracted one is *hotels*. A hard comparison gives zero similarity even if the words are very similar Therefore, we also use a *soft evaluation*[1] to provide better evaluation of the results. The soft evaluation relies on soft similarity measure between the two words. Among the different possibilities in [27], we use Levenshtein distance.

The results for the four datasets are summarized in Tables 2 to 5. For the English MACWorld dataset, D-rank and H-rank give the same F-score value 0.27 in the hard evaluation approach, which is 5% and 10% better than Cl-rank and TF method, respectively. D-rank provides better precision but worse recall comparing to H-rank.

1 http://cs.uef.fi/paikka/Radu/tools/SoftEval/

Table 2: Results for the MACWorld Dataset.

Hard Measures			
Method	Precision	Recall	F-score
D-rank	0.32	0.24	0.27
Cl-rank	0.26	0.20	0.22
H-rank	0.22	0.37	0.27
TF	0.16	0.18	0.17
Soft Measures			
Method	Precision	Recall	F-score
D-rank	0.58	0.46	0.50
Cl-rank	0.56	0.42	0.47
H-rank	0.48	0.68	0.55
TF	0.34	0.38	0.36

Table 3: Results for the Guardian Dataset.

Hard Measures			
Method	Precision	Recall	F-score
D-rank	0.24	0.37	0.29
Cl-rank	0.26	0.40	0.31
H-rank	0.20	0.29	0.23
TF	0.20	0.24	0.22
Soft Measures			
Method	Precision	Recall	F-score
D-rank	0.51	0.73	0.59
Cl-rank	0.54	0.75	0.62
H-rank	0.46	0.63	0.51
TF	0.52	0.63	0.57

Table 4: Results for Mopsi Services Dataset.

Hard Measures			
Method	Precision	Recall	F-score
D-rank	0.19	0.20	0.19
Cl-rank	-	-	-
H-rank	-	-	-
TF	0.11	0.26	0.12
Soft Measures			
Method	Precision	Recall	F-score
D-rank	0.42	0.45	0.43
Cl-rank	-	-	-
H-rank	-	-	-
TF	0.29	0.70	0.36

Table 5: Results for German Dataset.

Hard Measures			
Method	Precision	Recall	F-score
D-rank	0.26	0.27	0.23
Cl-rank	-	-	-
H-rank	-	-	-
TF	0.19	0.15	0.13
Soft Measures			
Method	Precision	Recall	F-score
D-rank	0.52	0.58	0.58
Cl-rank	-	-	-
H-rank	-	-	-
TF	0.37	0.32	0.29

In the soft evaluation approach, H-rank performs the best with 0.55, and D-rank is the second best method with 0.50 F-score value. Cl-rank and D-rank perform the best for another English dataset, Guardian, both in the hard and soft evaluations. Cl-rank gives slightly better F-score values in comparison with D-rank (about 2% and 3% for the hard and soft F-measure, respectively).

4 Conclusions and Future Work

We have introduced a new keyword extraction method from web pages, which is based on language and domain-independent features. Position of words in the page URL, title, hyperlinks, and headers in addition to word frequency are the information used for identifying the keywords. The method is fast and practical for webpages of different languages mainly because of avoiding complex NLP components. The results are promising comparing to other methods, which are usually slower and more complicated. We used Wikipedia to eliminate very common words, which are less likely the keywords. Several improvements can be added to the method, which are left as future research. Examples include a better weighting scenario for different word positions, removing similar keywords from the results, and adding more features from the webpage.

REFERENCES

[1] P D Turney (2003). Coherent keyphrase extraction via web mining. *ArXiv preprint cs*/0308033.

[2] P Tonella, F Ricca, E Pianta and C Girardi (2003). Using keyword extraction for web site clustering. In Fifth IEEE International Workshop on Web Site Evolution, pp. 41-48.

[3] M Grineva, M Grinev and D Lizorkin (2009). Extracting key terms from noisy and multi-theme documents. In Proceedings of the 18th international conference on World Wide Web, ACM New York, NY, USA, pages 661–670.

[4] W Yih, J Goodman and V R Carvalho (2006). Finding advertising keywords on web pages. In WWW 15th International Conference on World Wide Web, New York, NY, ACM, pages 213–222.

[5] R Mihalcea and P Tarau (2004). Textrank: Bringing order into text. In Proceedings of the conference on Empirical methods in Natural Language Processing.

[6] M Grineva, M Grinev and D Lizorkin (2009). Extracting key terms from noisy and multitheme documents. In Proceedings of the 18th International Conference on World Wide Web, pages 661–670.

[7] H Shah, M U S Khan and P Fränti (2019). H-rank: a keywords extraction method from web pages using POS tags. In IEEE International Conference on Industrial Informatics (INDIN), Helsinki.

[8] J K Humphreys (2002). Phraserate: An Html Keyphrase Extractor. Technical Report: University of California, Riverside, pages 1-16.

[9] D Kelleher and S Luz (2005). Automatic hypertext keyphrase detection. In IJCAI-05.

[10] A Onan, S Korukoğlu and H Bulut (2016). Ensemble of keyword extraction methods and classifiers in text classification. Expert Systems with Applications, Vol. 57, pp. 232-247.

[11] M Rezaei, N Gali and P Fränti (2015). Cl-rank: A method for keyword extraction from web pages using clustering and distribution of nouns. In International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), IEEE/WIC/ACM, pp. 79-84.

[12] Aditi Sharan and Siddiqi Sifatullah (2015). Keyword and keyphrase extraction techniques: A literature review. In International Journal of Computer Applications, 109.

[13] N Gali and P Fränti (2016). Content-based title extraction from web page. In International conference on web information systems and technologies (WEBIST), Vol. 2, pp. 204-210.

[14] D B Bracewell, F Ren and S Kuriowa (2005). Multilingual single document keyword extraction for information retrieval. In Proceedings of NLP-KE.

[15] M Litvak, M Last, H Aizenman, I Gobits and A Kandel (2011). DegExt—A language-independent graph-based keyphrase extractor. In Advances in Intelligent Web Mastering–3: Springer, pp. 121-130.

[16] S Gupta, G Kaiser, D Neistadt and P Grimm (2003). DOM-based content extraction of HTML documents. In proceedings of the 12th international conference on World Wide Web, ACM.

[17] M Litvak and M Last (2008). Graph-based keyword extraction for single-document summarization. In Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization, Manchester, pp. 17-24.

[18] C Florescu and C Caragea (2017). Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp. 1105-1115.

[19] G Turney (1999). Learning to extract keyphrases from text. Technical report, National Research Council, Institute for Information Technology.

[20] E Frank, G W Paynter, I H Witten, C Gutwin and C G Nevill-Manning (1999). Domain-specific keyphrase extraction. In 16th International joint conference on artificial intelligence, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, Vol. 2, pp. 668-673.

[21] S Brin and L E Page (1998). The anatomy of alarge-scale hypertextual web search engine. Comput.Netw.ISDN30, 107–117.

[22] Z Liu, P Li, Y Zheng and M Sun (2009). Clustering to find exemplar terms for keyphrase extraction. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Vol. 1, pp. 257-266.

[23] K L Poola and A Ramanujapuram (2009). Techniques for keyword extraction from URLs using statistical analysis. Patent Application Publication, Apr. 2, US, 2009/0089278 A1.

[24] A Bellaachia and M Al-Dhelaan (2012). Ne-rank: A novel graph-based keyphrase extraction in twitter. In Proceedings of IEEE CS.

[25] G Matošević (2015). Using anchor text to improve web page title in process of search engine optimization. In proceedings of the Conference on Information and Intelligent Systems, Varăzdin, Croatia.

[26] D R Radev, H Jing, M Sty and D Tam (2004). Centroid-based summarization of multiple documents. In Information Processing & Management, Vol. 40, pp. 919-938.

[27] N Gali, R Mariescu-Istodor, D Hostettler and P Fränti (2019). Framework for syntactic string similarity measures. Expert Systems with Applications, Vol. 129, pp. 169-185.