Keyword Extraction for Webpage Clusters

Vladimir Salin[1], Maria Slastihina[1], Ivan Ermilov[2], René Speck[2], Sören Auer[3], and Alexander Sytnik[1]

[1] Saratov State Technical University, 410054 Saratov, Russia,
salinvs@gmail.com, mashaslas@gmail.com, as@sstu.ru
[2] Universität Leipzig, AKSW/BIS, PO BOX 100920, 04009 Leipzig, Germany,
ivan.ermilov@informatik.uni-leipzig.de, speck@informatik.uni-leipzig.de
[3] Universität Bonn, CS/EIS, Römerstraße 164, 53117 Bonn, Germany
auer@cs.uni-bonn.de

Abstract. The volume of unstructured information presented on the Internet is constantly increasing, together with the total amount of websites and their contents. To process this vast amount of information it is important to distinguish different clusters of related webpages. Such clusters are used, for example, for template induction, keyword extraction, and recommendation algorithms. A variety of applications (such as semantic analysis systems, crawlers and search engines) utilize clustering algorithms to recognize thematically connected webpages. The clustering is performed based on sets of webpage features, for instance, hyperlinks between webpages, search logs or DOM tree structures. In this article we present an approach for keyword extraction, which utilizes two different clustering algorithms. The first algorithm is based on the analysis of the textual information, the second one – on the statistical information (obtained from the Google Analytics API). We evaluate our approach with three different **corpuses** of webpages.Also, we compare our keyword extraction approach with a non-clustered baseline and show a significant improvement in the relevance of extracted keywords.

1 Introduction

The volume of unstructured information presented on the Internet is constantly increasing, together with the total amount of websites and their contents. Finding information relevant to a problem at hand from such a vast amount of information is still a major challenge. The prevalent approach for facilitating the search for information on the Web is keyword search. Keyword-based search engines perform indexing and retrieval by extracting and associating a set of keywords with each webpage. However, in some cases the assignment of keywords to a webpage is not feasible due to the absence of sufficient textual data. In such cases a webpage can be grouped (i.e. clustered) together with related webpages containing textual data.

Webpage clustering is used in a variety of web data extraction applications, for example, for template induction, keyword extraction and recommendation algorithms. In this paper we consider webpage clustering as a constituent of our keyword extraction process. The majority of the keyword extraction systems assign keywords for each webpage separately. Thus, not all of the webpages related to a search query are detectable (e.g. a page containing only one figure), which leads to an incomplete result set.

To tackle the problem of detectability of such webpages we designed and implemented an approach, which clusters webpages and extracts keywords for these clusters. The rationale of our approach is that if we group webpages without sufficient textual content together with webpages containing sufficient textual content, we can assign relevant keywords to the whole group. Keywords for the group of webpages also identify each webpage in the group individually. Thus discoverability of webpages containing no textual data becomes possible through the keywords assigned to the whole cluster. For the clustering our framework utilizes two approaches: (1) based on the link graph extracted from the webpages, and (2) based on statistical access log data about the website (i.e. obtained from Google Analytics API). The first approach employs a graph clustering algorithm (e.g. BorderFlow) to obtain clusters from the link graph. The second approach clusters webpages based on which pages were visited by a user during one session and which keywords she searched for before navigating to the target webpage. After clustering we use Apache Lucene to extract keywords and demonstrate that our approach performs better than a non-clustered baseline.

In particular our contributions are:

- We design, describe and implement a webpage clustering algorithm based on statistical information about webpages.
- We implement a framework for keyword extraction, which can handle pages without textual content.
- We provide the source code of the whole framework for further reuse.

The paper is structured as follows: In Section 2 we outline existing web data extraction systems, techniques utilized in those systems as well as related work on web data extraction for ontology learning and graph clustering algorithms. In Section 3 we present the architecture of our keyword extraction framework. In Section 4 we discuss the crawling approach used in our framework. In Section 5 we introduce the two clustering approaches utilized in our framework. The keyword extraction mechanisms are described in Section 6. We evaluate our framework in Section 7 and conclude as well as outline future work in Section 8.

2 Related Work

The research field Web Data Extraction (WDE) is dealing with Information Extraction (IE) on the Web. An outline of existing WDE systems and their application domains is given in [7]. In particular, the authors distinguish enterprise and social web applications. This survey provides insights on the techniques for WDE as well as it gives examples for possible applications. However, it does not delve into the inner workings of WDE systems and thus lacks information on clustering approaches and their usage inside WDE systems.

[5] is another WDE work concentrated solely on an overview of WDE techniques. The authors differentiate two main types of techniques, one is based on wrapper/template induction and an another is automatic extraction. For the purpose of template induction clustering algorithms are used extensively. The main feature of clustering in this case is that clusters are disjoint (i.e. one page can not be in two different clusters at the same time). Obviously, only one template should be created for a particular page. In various systems clustering of webpages is based on different features. For instance, in [4] the authors utilize clustering based on paths in the DOM trees. The paths are identified by the analysis of a website and inferred from the search log. EXALG – an information extraction approach described in [1] – clusters webpages into "'equivalence classes"' by sets of tokens. EXALG represents an unsupervised template induction approach. The main difference between clustering used in above mentioned approaches and our work is that we utilize statistical access log data as, for example, produced by Google Analytics for the clustering algorithm.

Another related field of research is IE for Ontology Learning (OL). In [15] Suchanek describes IE from well-formed sources, that are projects like DBpedia [2], YAGO [16] and KOG [20]. All three projects aim to extract information from Wikipedia and to construct an ontology, which supports querying (e.g. SPARQL queries) and question answering.In the same paper Suchanek reports about systems, which are designed to extract information from any web page on the Web: OntoUSP [14], NELL [3] and SOFIE [17]. These systems are centered around NLP processing of web documents using different approaches, but do not perform webpage clustering.

The following keyword extraction tools are related to our work. GenEx [18], a genetic algorithm for keyword extraction, is a rule-based system with 12 parameters and has been trained among others on websites. KEA [9], a keyphrase extraction algorithm, is based on machine learning with Naïve Bayes and can be trained faster than GenEx. It uses stopwords, stemming and three features for presenting keywords. It has been shown, that there is no significant difference between GenEx and KEA. KEA was extended in many ways. In [19] and [10] with website features. Maui [13] enhances KEA with semantic knowledge from Wikipedia, new features and a new classification model. DKPro [6] is a collection of software components for keyword extraction based on UIMA [8]. It selects, filters, ranks and evaluates keywords employing state-of-the-art approaches.

3 Architecture Overview

The architecture of our solution is based on the rationale that clustering can improve the relevance of extracted keywords.The majority of the keyword extraction methods from unstructured content employ a direct text analysis that computes the frequency of word occurrence in the document. For instance, in [11] each word is represented as vector whose values reflect the word occurrence frequency in a particular document. It is widely accepted that the F-measure for statistical data extraction methods increases with the size of the corpus. Thus to improve the quality of results it is important to analyse all related information sources instead of processing them separately.

The data extraction process is even more complicated due to the variety of features of the information representation on the Web. On the one hand HTML markup with the textual content provides us with a corpus, which is easy to process with statistical data extraction methods. On the other hand various types of information within webpages (e.g. figures, SVG documents, videos) require additional processing before the statistical data extraction methods can be applied. The following includes examples of the webpages, which contain information not suitable for statistical data extraction methods.

1. Webpages with scarce textual content insufficient for the extraction of keywords, which can describe this content with desirable precision.
2. Webpages with non-textual data, which requires additional processing (e.g. figures, SVG documents, videos).
3. Webpages with binary data (e.g. PDF, word processor documents, spreadsheets), which are represented in human-readable form, but are difficult to analyze programmatically.

In general, in order to extract data from such webpages it is necessary to either utilize its semantics (i.e. HTML markup information) or to aggregate the webpages for further analysis.

Overall there exist two different classes of approaches for the webpage processing with scarce information. The first class utilizes HTML markup information together with the contents of a webpage and thus is not suitable for webpages without these elements (e.g. webpages with only figures, videos or PDF files). In this paper we consider the second class, which preprocesses webpages by clustering them. The analysis of such clusters produces results relevant to the cluster itself as well as to every single page in particular.

In order to leverage webpage clustering, we designed the architecture of our approach according to the following modules (see Figure 1):

1. The *data collection module* crawls the website from an user-defined URL. The crawling process is further described in Section 4. The output of this module is a directed weighted link graph.
2. The *website clustering module* clusters the graph obtained by the data collection module. It implements the following clustering approaches (see Section 5):
   – Website link graph representation clustering
   – Statistical access log clustering
3. The *data extraction module* evaluates the clusters of webpages from the second module. The keywords relevant to each webpage in a cluster are extracted.

As a result our approach extracts keywords relevant for whole groups of webpages. These cluster keywords can be combined with keywords relevant to each individual webpage.

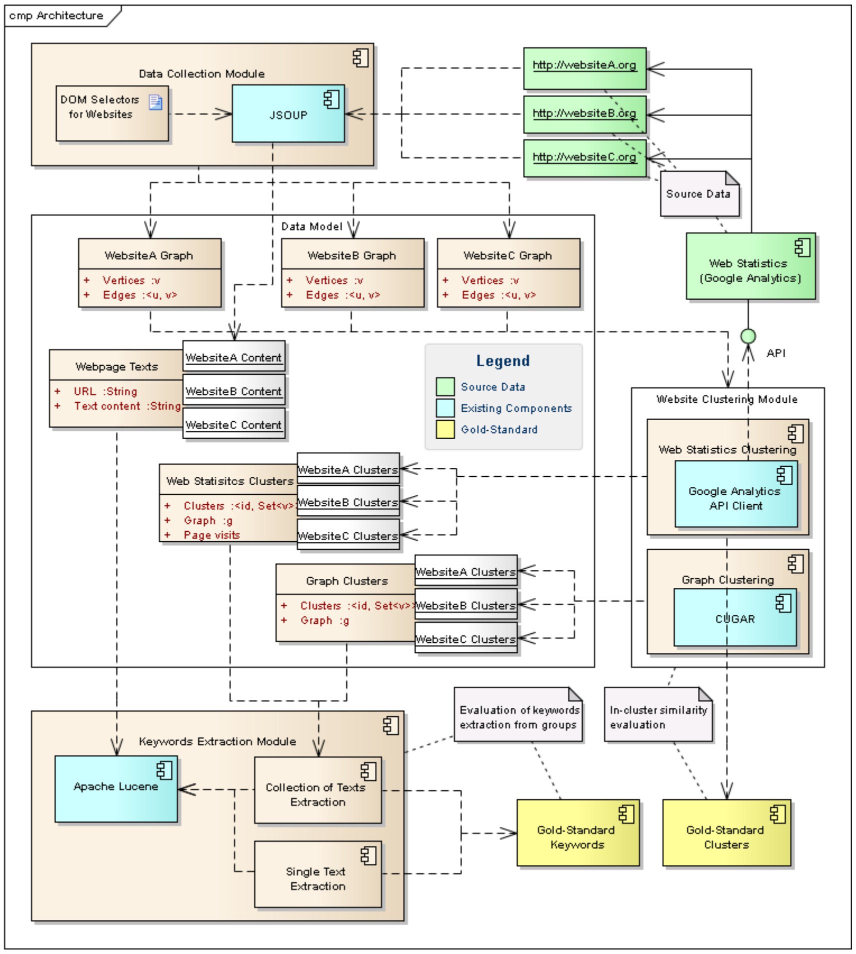

Fig. 1: The architecture of the keyword extraction process.

4 Crawling

The data collection module crawls a selected website and extracts the following types of information: (1) each webpage URL, (2) links between webpages, (3) webpage content. All the information is extracted using the JSOUP HTML parser. For correct usage it is necessary to construct a special configuration file for each processing website.

5 Clustering Approaches

For the website clustering we utilize two approaches to group webpages by using their metadata.The first approach processes a graph model of the website hypertext link structure. The second approach uses statistical access log data

```
input  : website URL
output : Graph G (Set N {nodes}, Set E[] {edges})
Initialize HashMap PagesParsed;
Initialize Stack URLs;
Push URL to URLs;
while sizeof URLs > 0 do
    currentUrl := pop URLs;
    Initialize HashSet Links;
    Links := (get all hyper references from :currentUrl);
    for each ref in Links do
        if !(URLs contain ref) and !(PagesParsed contain ref) then
            Push ref to URLs;
        else

        end
    end
    Put currentUrl, Links to PagesParsed;
end
Initialize Set N {nodes};
Initialize Set E[] {edges};
k := 1;
for i := 1 to sizeof PagesParsed do
    N[i] := Key of PagesParsed[i];
    Initialize Set Ntarge;̂Ntarget := Value of PagesParsed[i];
    for j := 1 to sizeof Ntarget do
        E[k][1] := Key of PagesParsed[i];
        E[k][2] := Ntarget[j];
        k := k + 1;
    end
end
Add nodes N to G;
Add edges E[] to G;
return G;
```

Algorithm 1: Constructing website graph from hypertext

of user visits. Both approaches are based on semantic similarity of webpages employing links and co-visitation as indicators for semantic similarity.

5.1 Graph Clustering

In this section we propose the graph clustering approach underlined by the following hypothesis: websites have groups of pages with a higher level of semantic similarity inside the group than with pages outside. Such groups of webpages are forming sections of websites referring to certain topics.

The graph clustering approach is based on the link graph model. Any website can be represented as an unweighted directed graph $G = (V, E)$, where:

- V is a set of vertices $v_1, v_2, \ldots, v_n \in V$, which correspond to webpages.
- E is a set of edges between vertices: $e(v, u) \in E, v, u \in V$, which are hyperlinks between webpages.

Although weights are not required for the most of the clustering algorithms, the usage of weights can drastically improve the clustering results. Therefore, for each graph edge we can calculate weights by computing the mean value of degrees for neighbor vertices.

After modeling the website structure as weighted directed link graph we can apply graph clustering algorithms to detect vertices coupled with each other. Every cluster calculated by the algorithms is associated with a group of corresponding webpages, which are analyzed together.

A straightforward approach for the clustering is to utilize existing graph clustering algorithms. The CUGAR Graph Clustering and Visualization Framework[4] includes the following five algorithms: Affinity Propagation, BorderFlow, Chinese Whispers, k-Nearest Neighbors and Markov Cluster Algorithm. This library allows to use these algorithms through an API without heavy interfaces and visualizations.

```
input  : Graph G, t {threshold}
output : Set C {clusters}

Initialize Set E[] {edges};
E[] := (get Edges from G);
Initialize Set W {edge weights};
for each e[] in E[] do
    n1c := Connectivity of e[1];
    n2c := Connectivity of e[2];
    w := (n1c + n2c) / 2;
    Add w to W;
end
Initialize File F;
for i := 1 to sizeof E do
    Initialize Triplet T weighted edge;
    T[1] := E[i][1];
    T[2] := E[i][2];
    T[3] := W[i];
    Add T delimited with commas to F;
end
Initialize HashTable C {Clusters};
Initialize BorderFlow BF with file F and threshold t;
C := (cluster F with BF);
return C;
```

Algorithm 2: Clustering website graph with BorderFlow

The graph structure analysis approach can be described by the following steps:

[4] `http://sourceforge.net/projects/cugar-framework` by R. Speck and A.-C. Ngonga Ngomo (University of Leipzig)

1. *Re-engineering website graph model.* For a given website our parsing subsystem[5] crawls through each internal hyperlink to create the link graph model of the website. As a crawling strategy our parsing subsystem implements non-recursive Depth-First Search (DFS).When graph traversing is complete the reconstructed graph model is stored in the memory for later analysis.
2. *Exporting data.* We extract the information from the database and generate a CSV file representing the graph of the chosen website. The CSV format has been chosen for graph export as the most lightweight format supported by CUGAR. If required, weights are calculated as described above and provided along with the edges.
3. *Clustering a graph.* A clustering algorithm is executed with the website graph as input data. To perform clustering with CUGAR, we have wrapped its API with a command-line Java application[6] to easily execute any algorithm supported by CUGAR. When the clustering is complete, we parse the CUGAR output file and store the results in the database. The results contain groups of pages clustered by a certain criterion (depending on the chosen algorithm). Finally, we store a set of these groups as well as their members, corresponding to the website's pages. The grouping results are further used for keyword extraction.

It is also important to note that other clustering algorithms than the ones provided by CUGAR can be employed as well. For example, we have experimented with TopGC [12] and the results of this experiment are comprised in Section 7.

5.2 Statistical-based Clustering

Our second approach exploits the user behavior on a website. In general, there are the following three methods to group webpages using access log statistics:

1. *Most visited webpage analysis.* This analysis is based on the rationale that the users know what pages are the most interesting and are more relevant to the website's topic than others. These pages can be analysed together.
2. *Visited during one session webpage analysis.* This analysis assumes that the user is interested in one topic (or several thematically connected topics) during one browsing session. That is why the webpages visited by a user during one session can be combined into one group.
3. *Search query analysis.* This analysis is based on the rationale, that a webpage visited after a web search is related to the search keyword originally entered by the user. We assume that the webpages visited by the user during this session are thematically connected with each other and the entered search keyword.

In this paper, only the method for grouping webpages visited during one session, is discussed in detail and evaluated. By entering a website, the user is

[5] `https://github.com/sainnr/pagegroups/tree/master/parser`
[6] `https://github.com/sainnr/pagegroups/tree/master/core`

looking for a certain kind of information, which he is currently interested in. We assume that webpages visited by the user during one session correspond to similar topics. We also exclude webpages, which are skipped by the user and visited less then few seconds. The resulting set of pages can then be analyzed together.

To track user visits and routes, website access log statistics gathering tools can be used. In this paper, we have used the Google Analytics service to gather website statistics. Other statistical applications of such kind can be used if they allow to identify pages, visited during one session.

The following workflow describes this approach:

1. Choose a landing page URL, from which the user starts the session.
2. Request webpages, visited consequently after the landing page during the same session.
3. Filter results with a threshold for the page view time to exclude pages skipped by the user.
4. Combine all resulting pages starting from the chosen landing page in a group.
5. Repeat steps 1-4 for every webpage of the website.

```
input  : website URL, startDate, endDate
output : Set C {clusters of webpages}
Connect and authorize to Google Analytics;
Initialize HashSet LP {Landing Pages};
LP := (get Landing Pages for :URL between :startDate and :endDate);
Initialize HashTable C {Clusters};
for each p in LP do
    Initialize HashSet NP {Next Pages};
    NP := (get Next Pages visited after :p);
    for each np in NP do
        pageViews := (get count of page views for :np);
        timeOnPage := (get time spent on page :np);
        if $pageviews < 2$ or $timeOnPage < 5$ then
            Remove np from NP;
        end
    end
    Add p, NP to C;
end
return C;
```

Algorithm 3: Obtaining clusters from web statistics

As the result, we obtain a set of groups, whose elements correspond to the pages, visited together by website users.

6 Keywords Extraction

Apache Lucene is a high-performance, open source library for implementing full-text search applications. This library provides a vast amount of search and indexing algorithms suitable for nearly any related application. Our implementation uses Apache Lucene to extract keywords from webpage clusters. The keyword extraction module performs the following steps:

- We assume that the webpages of the chosen website were already clustered by at least one of the clustering algorithms.
- Each cluster is an indivisible item and is used as an input for the Apache Lucene keyword extraction.
- Apache Lucene returns the set of keywords relevant to the webpages cluster.

The F-measure of the keyword extraction algorithm is influenced by the choice of clustering algorithm as we show with our evaluation.

7 Evaluation

To evaluate the two grouping approaches, described above, we have used the following techniques:

1. Estimate a semantic similarity of webpages in a group, by comparing clusters with a gold-standard.
2. Extract keywords from single webpages and from the cluster of webpages and compare them with the gold-standard.
3. Evaluate the precision of keywords extracted for webpages without text content.

The first evaluation method includes a large amount of manual work to review every cluster and every webpage from all found clusters. However, it produces high quality results, because we can manually estimate a human-readable content of the webpage and decide about its similarity with other webpages.

The other two methods are based on keywords extraction performed for all pages in the cluster. The second technique allows us to automatically evaluate the quality of grouping: webpages with similar semantics would have a higher occurrence rate for domain-specific terms. While single webpages analysis would produce uniform distribution without high-rated domain terms.

Using the third technique, we are able to compare keywords for pages with no text content. As in the previous method, we extract keywords for the whole cluster and propagate them for every webpage in the cluster. If a cluster contains webpages without text content, such webpages will obtain keywords as well. Finally, we compare keywords of the webpages without text content with the gold-standard.

7.1 Gold Standard

To evaluate the precision of clustering approaches and keywords extraction results, we need to define a gold standards for both.

For the clustering evaluation, we have manually defined a set of possible clusters of webpages, where each cluster corresponds to a certain topic. For example, it can be a cluster of university department pages, or a cluster of news articles for the last month. To evaluate the precision of the clustering approaches, we count the intersection of pages in a computed cluster with the respective gold-standard cluster. The median intersection count for all found clusters represents the precision of clustering approach.

To obtain a gold-standard for keyword extraction, we asked experts to go through all webpages of the website and define a set of keywords for them. To evaluate the precision, we calculate the occurrence rate of automatically found keywords in the gold-standard. The percentage of keywords from the gold-standard covered by found keywords, describes the quality of the approach.

7.2 Data Sources

To perform evaluation, we use three real-world websites as data sources. Two of them have Russian language content, another one has content in English. It is required to have a access log statistics gathering system (i.e. Google Analytics) being connected to the website and all three websites meet this requirement.

The first website is the website of the Saratov State Technical University[7]. It runs the Drupal CMS with custom templates and is connected with Google Analytics since more than two years. The website contains information about different topics, including faculties, departments, institutions of the university and has overall more than 12,000 webpages.

The second website is the website of the educational center 'Virtual branch of Russian Museum[8]'. It provides information about educational and cultural programs for children and adults in the area of arts and history. The website is also connected to Google Analytics and has about 1,500 webpages.

The website of Agile Knowledge and Semantic Web (AKSW) Research Group[9] provides information about AKSW research group, their projects, team members, events, etc. It has about 350 pages available on the main domain. Also, there is a set of OntoWiki webpages, which mostly contain technical and administrative information and were thus not analyzed.

7.3 Experimental Setup

The processes of website crawling, clustering and keywords analyzing were performed on quadcore Intel(R) Xeon(R) CPU E5-2670 v2 with 2.50GHz and 64Gb of total RA memory. It runs under Oracle Linux Server release 6.4 with OpenJDK Runtime Environment (IcedTea6 1.11.11.90), used for Java applications.

7.4 Experimental Results

The web crawling results are represented in Table 1.

[7] `http://www.sstu.ru`
[8] `http://museum.seun.ru`
[9] `http://aksw.org`

	sstu.ru	*aksw.org*	*museum.seun.ru*
Num. of vertices	11,987	563	1,780
Num. of edges	562,996	12,103	47,193
Conn. rate	46.96	21.49	26.51
Time, sec.	20 573	934	3 190

Table 1: Website crawling results

For data clustering we selected a sample containing 100 webpages from AKSW, education center websites and 500 webpages from SSTU website. The sample selection is based on the size of the gold standard for the respective websites. As can be seen in Table 2 the clustering using access log statistics has incomplete page coverage. This is attributed to the fact, that for some webpages statistics are not available and some webpages we filter out (e.g. webpages visited only once). We assess the semantic similarity of the clusters manually by selecting a sample of 30 clusters and present the results in Table 3. The high semantic similarity for statistics-based clusters is attributed to the features used in the algorithm.

		sstu.ru	*aksw.org*	*museum.seun.ru*
Graph clustering (BorderFlow)	Num. of clusters	665	14	103
	Avg. size	17.5	26.86	42.17
	Max. size	531	82	42.17
	Pages covered	11,801 (98.4%)	376 (66.7%)	1,664 (93.4%)
	Time, sec	16,010	45	1,382
Graph clustering (access log statistics)	Num. of clusters	44	36	5
	Avg. size	32.5	27.36	33.4
	Max. size	152	150	112
	Pages covered	573 (4.7%)	195 (34.6%)	254 (14.2%)
	Visits per last month	13,721	4,654	314
	Time, sec.	43	32	29

Table 2: Graph clustering results.

	sstu.ru	*aksw.org*	*museum.seun.ru*
Graph clusters	93.3%	92.8%	90%
Statistics-based clusters	100%	100%	100%

Table 3: Semantic similarity of webpages in clusters.

We extract the keywords for the each single webpage (see Table 4) and for clusters of webpages (see Table 5). The unique keywords within the clusters of webpages are less likely to be found. Due to this fact the total number of keywords for clusters is drastically lower. We compare the extracted keywords with the gold-standard and calculate the F-measure for the keyword extraction involving each of the clustering algorithms (see Table 6). Recall is the intersection between keyword sets from the gold standard and obtained by the keyword extraction process. Precision is defined as percentage of keywords from the gold standard in the overall number of keywords found. As can be seen in the results the F-measure for the clustering-based keyword extraction is substantially higher than for single page keyword extraction.

Also we were able to assign keywords for the webpages without textual content. 77 (0.6%) from SSTU, 8 (1.4%) from AKSW, and 39 (2.2%) from museum.seun.ru websites.

	sstu.ru	*aksw.org*	*museum.seun.ru*
Num. of webpages w. text	11,937 (99.5%)	561 (99.6%)	1,693 (95.1%)
Avg. text length, chr.	3,420	1,972	3,351
Total keywords	1,290,664	62,556	251,497
Avg. keywords per webpage	116.2	140.9	150.8
Time, sec.	24,305	196	1,963

Table 4: Keywords extraction results for single webpages.

8 Conclusions and Future Work

This article describes clustering-based keywords extraction framework. The framework is able to extract keywords for the webpages without textual content. The main contribution is the implementation of the statistical webpages clustering algorithm, which improved the performance of keywords extraction algorithm for the webpages without textual content.

Our clustering-based keyword extraction framework still has the following limitations:

1. The data collection module works only for websites with unified structure and it is necessary to implement some structure details into data collection module configuration file.
2. The statistical clustering module works only for websites with access log information.
3. The percentage of webpages covered by the clustering module is not yet sufficient.

In further work we will extend the framework in two directions: mitigate limitations and applying the framework as a backend for existing applications (e.g. conTEXT[10]).

References

1. A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 337–348. ACM, 2003.

[10] `http://context.aksw.org/app/`

Graph clustering			
	sstu.ru	*aksw.org*	*museum.seun.ru*
Total keywords	753,869	12,704	153,999
Avg. keywords per cluster	1,137	907	1,495
Time, sec.	3,660	46	540
Statistics-based clustering			
Total keywords	58,819	17,071	2,495
Avg. keywords per cluster	1,434	531	831
Time, sec.	278	51	12

Table 5: Keywords extraction results for clusters of webpages.

Single pages extraction			
	sstu.ru	*aksw.org*	*museum.seun.ru*
F-measure	6.1%	6.2%	4.8%
Precision	3.5%	3.3%	2.8%
Recall	69.4%	91.1%	67.9%
Keyword extraction for BorderFlow			
F-measure	28.5%	33.5%	20.8%
Precision	28.33%	30.9%	16.9%
Recall	31.9%	38.4%	36.6%
Keyword extraction for statistical-based clusters			
F-measure	19.2%	11.8%	13.3%
Precision	13.9%	7.1%	7.7%
Recall	46.8%	33.3%	50%

Table 6: F-measure, precision and recall for keyword extraction.

2. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007.
3. A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka, Jr., and T. M. Mitchell. Coupled semi-supervised learning for information extraction. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 101–110, New York, NY, USA, 2010. ACM.
4. D. Chakrabarti and R. Mehta. The paths more taken: matching dom trees to search logs for accurate webpage clustering. In *Proceedings of the 19th international conference on World wide web*, pages 211–220. ACM, 2010.
5. K. Devika and S. Surendran. An overview of web data extraction techniques. *International Journal of Scientific Engineering and Technology*, 2(4), 2013.
6. N. Erbs, P. B. Santos, I. Gurevych, and T. Zesch. Dkpro keyphrases: Flexible and reusable keyphrase extraction experiments. *ACL 2014*, page 31, 2014.
7. E. Ferrara, P. D. Meo, G. Fiumara, and R. Baumgartner. Web data extraction, applications and techniques: A survey. *CoRR*, abs/1207.0246, 2012.
8. D. Ferrucci and A. Lally. Uima: An architectural approach to unstructured information processing in the corporate research environment. *Nat. Lang. Eng.*, 10(3-4):327–348, Sept. 2004.
9. E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. Domain-specific keyphrase extraction. In *Proceedings of the 16th International*

Joint Conference on Artificial Intelligence - Volume 2, IJCAI'99, pages 668–673, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

10. D. Kelleher and S. Luz. Automatic hypertext keyphrase detection. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05, pages 1608–1609, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.
11. C.-H. Lee and H.-C. Yang. A web text mining approach based on self-organizing map. In *Proceedings of the 2Nd International Workshop on Web Information and Data Management*, WIDM '99, pages 59–62, New York, NY, USA, 1999. ACM.
12. K. Macropol and A. Singh. Scalable discovery of best clusters on large graphs. In *Proceedings of the VLDB Endowment*, volume 3, pages 13–17. VLDB Endowment, 2010.
13. O. Medelyan, E. Frank, and I. H. Witten. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1318–1327, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.
14. H. Poon and P. Domingos. Unsupervised ontology induction from text. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 296–305, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.
15. F. Suchanek. *Studies on the Semantic Web*, chapter Information Extraction for Ontology Learning. Akademische Verlagsgesellschaft - AKA GmbH, P.O. Box 41 07 05, 12117 Berlin, Germany, 2014.
16. F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007.
17. F. M. Suchanek, M. Sozio, and G. Weikum. Sofie: a self-organizing framework for information extraction. In *Proceedings of the 18th international conference on World wide web*, pages 631–640. ACM, 2009.
18. P. D. Turney. Learning algorithms for keyphrase extraction. *Inf. Retr.*, 2(4):303–336, May 2000.
19. P. D. Turney. Coherent keyphrase extraction via web mining. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 434–439, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.
20. F. Wu and D. S. Weld. Automatically refining the wikipedia infobox ontology. In *Proceedings of the 17th international conference on World Wide Web*, pages 635–644. ACM, 2008.