Web indexing and search with local language support

Damir Krstinić, B.Sc. and Ivan Slapničar, Ph.D.
Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture
University of Split
R. Boskovica b.b., 21000 Split, Croatia
E-mail: damir.krstinic@fesb.hr, ivan.slapnicar@fesb.hr
Phone: +385 21 305 617

Abstract— **Web search is becoming essential for every day life, where major need arises for extracting relevant knowledge from enormous amounts of the available data. In a modern information retrieval systems, data is modeled as a term-by-document matrix. User query is represented as a vector and database search becomes a simple vector operation. The Latent Semantic Indexing (LSI) method reduces the size of term by document matrix and improves the performance of information retrieval system. Great majority of these systems are based on the English language. Although these systems are applicable to documents in other languages, they can suffer from incomplete terms recognition. We focus on languages with a complex set of grammar rules where improvement can be achieved by giving the indexing system basic knowledge of the language, and ability to recognize different forms of the same word. Using this technique, original matrix can be reduced by order of magnitude and important term-document connections strengthened. We are developing web indexing engine with local language support using Ispell dictionary files. As part of this effort, Croatian language dictionary files have been developed.**

I. Introduction

THE evolution of the digital libraries and the Internet has produced enormous amounts of the data covering almost any field of a human interest. In this situation, major need arises for extracting relevant knowledge form all of the data available, especially when data is distrubuted across a resources over the network. To provide users with the tool for locating and extracting relevant data, number of techniques were developed for automatic indexing of the textual materials. Recently developed techniques are based on the concept of a vector space. Data is modeled as term-by-document matrix, where each column of the matrix represents one document, with nonzero values representing terms found in that document. User query is represented as a vector, and the database search becomes a simple vector operation. As the size of this matrix depends on the number of the documents indexed and the number of terms found in all documents, large storage space and great computational power is necessary to process these matrices. The Latent Semantic Indexing method (*LSI*, see [1], [2], [3]) reduces the size of the matrix by exploiting higher order structure of the terms by documents connections, thus improving the performance of the information retrieval systems.

The term dimension of the matrix depends on the number of the different terms found in the processed documents. In the process of data indexing, each new term found generates new row in the matrix. In the languages with a complex set of grammatical rules, same word could have many different forms, resulting in several rows of the matrix representing the same word. As a result, additional computational power and storage space is required. Beside that, not all of the documents relevant to the user query will be spotted because system will recognize only those documents in which the words from the query appear in the same form as in the query itself.

By giving the indexing system basic knowledge of the human language, and ability to recognize different forms of the same word, the original matrix can be reduced by order of magnitude, and important term-document connections strengthened. We are developing web indexing engine focused on languages with complex set of grammar rules where standard English-based web indexing systems have low performance due to complex structure of the language.

The remaining of the paper is organized as follows: The following section gives a short review of distributed data indexing techniques. In section 3 problems with grammatical structure of the language are explained with several examples in different languages. We are presenting our proposal for solving the problem with local language in section 4, followed by the conclusion and ideas for future work in section 5.

II. Review of web indexing techniques

IN the vector space data model, the vector is used to represent each document in the collection. Each component of the vector is associated with the particular term, where nonzero value represents occurrence of the particular term in the document. These values are often weighted to emphasize the importance of distinguishing terms. Document vectors are stored as columns of the term-by-document matrix. For large document collections, each document generally uses only a small subset of the entire dictionary of terms generated for a given database, and most of the elements of a term by document matrix are zero.

Query matching is finding the documents most simi-

lar to the query in use and weightening of terms. Geometrical interpretation of query matching is finding the document vectors closest to the query according to some measure.

A. Web spider

BEFORE processing user queries, an index of all available documents must be generated where each record represents information about one document. This is the most computationally and memory consuming process, and the quality of the complete information retrieval system strongly depends on accuracy and completeness of the document index. This job is performed only once, so it must be carefully prepared. The document index can later be updated, both by adding new documents to the index, and by updating existing documents. Process of generating document index could be itself divided in two steps: document collecting and analyzing, and generating document matrix by performing *SVD* decomposition ([3], [7]). Programs designed for collecting information about available documents are often called *Web Spiders* or *Web Crawlers* ([4]).

In a distributed environment like Internet, documents are spread over many network resources. In this case, list of all available documents do not exist, nor could be easily acquired. Documents are organized in a way that every document may point to other documents, and every document could be pointed to by many other documents, where multiple loops are introduced. Main task of the spider is to analyze the structure of the *HTML* document and create document index record based on the terms found in the document. Additionally, number of other documents with links pointing to the document is taken into consideration. Documents pointed by many other documents are considered more relevant, and weights are adjusted to make them appear closest to the user query. List of detected links is extracted for each analyzed document, and new documents are added to the list of available documents. If indexing is limited to a specific domain, only links pointing to the documents in appropriate domain are added to the list. Each term found in the document is checked against the dictionary. If term is not found, new entry is added to the dictionary. Some terms are very frequent in human language (terms like: is, this, are, the, on). These terms are skipped because they have no distinguishing meaning.

B. Latent Semantic Indexing

COMMON measure of a similarity of the user query and the documents in the list is the cosine of the angle between the query and document vectors. If A is $t \times d$ matrix, where t is number of terms, and d is number of the documents, documents are coded in columns $a_j, j = 1, ..., d$. Cosines are computed according to the formula:

$$cos\theta_j = \frac{a_j^T q}{\parallel a \parallel_2 \parallel q \parallel_2} = \frac{\sum_{i=1}^{t} a_{ij} q_i}{\sqrt{\sum_{i=1}^{t} a_{ij}^2} \sqrt{\sum_{i=1}^{t} g_i^2}} \quad (1)$$

In the large document collections both documents and query are using only a small subset of the whole dictionary of terms. Matrix A, and query vector q are very sparse, and the vectors describing documents and query are wildly spread in the t-dimensional space spanned by all terms used in all of the documents in the collection.

The technique of *Latent Semantic Indexing* ([1], [2], [3]) uses truncated Singular Value Decomposition to project very high dimensional document and query vectors into a low dimensional space. Briefly,

$$A = U \Sigma V' \approx U_k \Sigma_k V_k' \quad (2)$$

where A is original $t \times d$ matrix, U is $t \times d$ with orthonormal columns, V is $d \times d$ with orthonormal columns, and Σ is diagonal with the diagonal entries sorted in decreasing order. Matrix A is approximated with product $U_k \Sigma_k V_k'$ where $U_k \equiv U(u_1, u_2, ..., u_k)$ are first k columns of U, $V_k \equiv V(v_1, v_2, ..., v_k)$, and $\Sigma_k \equiv diag(\sigma_1, \sigma_2, ..., \sigma_k)$

Details of the above process can be found in ([7]). Projecting high dimensional vectors onto a low dimensional space has an effect of grouping similar vectors. These similarities are not visible in the original high dimensional vector space spanned bay all terms found in the entire collection of the documents. User query must be projected onto the same subspace to be compared with the document vectors. Additionally, lowering the rank lowers the computational cost of the query matching and memory for storing document vectors.

TABLE I
NOUN **treaty** - ENGLISH LANGUAGE

query	Num. doc.
treaty	3 300 000
treaties	1 390 000
treaty AND treaties	409 000

III. LANGUAGE GRAMMAR

FOR our experiments, we have designed an experimental spider ([4])in *Java* [11] programming language. Spider was started with empty dictionary, and

TABLE II

VERB **compute** - ENGLISH LANGUAGE

query	Num. doc.
compute	2 650 000
computed	2 880 000
compute AND computed	658 000

TABLE III

NOUN **zahl** - GERMAN LANGUAGE

query	Num. doc.
zahl	2 040 000
zahlen	2 520 000
zahl AND zahlen	428 000

indexing process was limited to one Croatian newspaper domain where all of the documents are expected to be in Croatian language. Same experiment was repeated with several others media sites, and some official sites. After indexing selected domains, number of terms to number of documents ratio was around 10^2.

By inspecting dictionary automatically generated in the indexing process, number of entries were found to represent a same word in different forms, due to complexness of Croatian language grammar. Effect of this was that the dictionary was several times larger than it is necessary, consuming more memory and computational power, both in the phase of analyzing documents, and in the computation of truncated *SVD* decomposition. Additionally, having same word represented in different forms results in loss of important semantic connections. By projecting full term-by-document matrix on the lower dimension subspace, these semantic connections were partially restored, but drawback in performance of query matching remains.

To check our hypothesis of necessity of including grammar rules into web indexing and search process, we conducted a series of experiments with *Google* search engine ([8]) using queries in English, Croatian and German language. Results of these experiments are given in tables (I) to (VIII).

In table (I) results of querying Google with English noun *TREATY* in two different forms are given. In the first case, only documents containing singular are found, and in the second case only documents containing plural are found. Query containing both forms of the same word returns three to eight times lower number of documents comparing to queries containing only one form. In table (II) results of a similar experiment with verb *COMPUTE* are given. Ratio of documents found when both forms of the verb are included in the query compared to the query with one form is slightly above 4. In table (III) results are given for German noun *ZAHL* in singular and plural. Results are similar to those of the experiments with the English language words. These examples confirm that conventional search engine will not find all of the documents relevant to the user query. For example, document in which some noun is used in plural will not be considered relevant to the query with the same noun in singular.

TABLE IV

ENGLISH PHRASE

query	Num. doc.
indexing and searching distributed data	97 300
distributed data index and search	1 190 000

In our next experiment, we have compared two queries consisting of the basically the same phrase, slightly different in the form of the verbs used. Results of this experiment are in table (IV). As two word are used in different forms, number of documents returned for the first query is more than ten time lower than number of documents returned for the second query. This difference in number of documents found is emphasized by the fact that phrase used in first query is more "natural", in the form that will be found in the spoken language and probably used by regular user.

TABLE V

VERB **programirati** - CROATIAN LANGUAGE

query	Num. doc.
programirati	2 290
programiran	806
programirati AND programiran	55

Results of the experiments with queries using single Croatian language word are given in tables (V) and (VI). In table (V) verb *PROGRAMIRATI* (*to program* in English) is used in two different forms. Due to complexness of Croatian language grammar, where verb can have up to 30 different forms, one query returns approximately three times more documents than another. Query with both forms of the verb returns more then 40 times less documents than query with one form of the verb. Similar situation is with the noun *VOZAČ* (*driver* in English), given in table (VI).

In tables (VII) and (VIII) results of querying *Google* with two croatian phrases are given. In both cases, first query with less documents found is in the form that it would be found in the spoken language for the given phrase. In the example (VIII) 60 times more documents is found for query where word from the phrase are used in their basic form, and only one document is found for the query with all terms from both queries. It is obvious from this results that only a subset of

TABLE VI
NOUN **vozač** - CROATIAN LANGUAGE

query	Num. doc.
vozač	132
vozači	53
vozač AND vozači	12

TABLE VII
CROATIAN PHRASE

query	Num. doc.
ponuda apartmana na otoku Hvaru	87
apartmani ponuda otok Hvar	289

the complete set of the documents relevant to the user query is returned, depending on the form of the words used in the query. Completeness of this subset depends on the complexness of the language grammar. Results are better for the languages with relatively simple grammar, like English, and poor for the languages with complex grammar, where only a small subset of the relevant documents is returned. Situation gets worst for the queries with more than one word, where only those documents containing each word from the query in the exact form as in the query are returned.

IV. Adding local language support

AS we have shown by the previous analysis, in order to improve the performance of the Information Retrieval (*IR*) system, basic knowledge of the language grammar should be implemented. This is especially important for the languages with complex set of grammar rules where significant improvement can be achieved. Even for English language, part of the documents relevant to the user query is missed because of the language structure.

After considering different possible forms of language syntax implementation, we decided to create web indexing engine based on *Ispell* [10] dictionary files. *Ispell* is an interactive spell-checking program for Unix. *Ispell* dictionary and grammar rules files are available for great number of languages widely used on the Internet. Additionally, open structure of *Ispell* which allows implementation of new languages was main reason for selecting this format. This way, *IR* engine can load appropriate dictionary and automatically be configured for particular language, resulting in better performance than standard systems.

TABLE VIII
CROATIAN PHRASE

query	Num. doc.
ronilački turizam	10
ronjenje turizam	625
(all words)	1

In the process of the creation of the document index, each term found in the document is transformed into its basic form and indexed according to the dictionary, or added to the dictionary if not found. By using this technique, resulting dictionary is smaller, and document index more consistent. Each term from the user query is also transformed to its basic form.

In collaboration with the Croatian National Corpus Project ([5], [6]) we have developed Croatian language dictionary files. Beside their implementation in web search engine, the created language files can be used for spell-checking with *Ispell*, and will be publicly available in their final form. Grammar rules for different types of words are defined, together resulting in more than 100 different rules. Each word in the dictionary is written in its basic form and marked by a tag representing appropriate rule for generating all forms of the given word. Dictionary with 20000 marked words has been created, including 12000 nouns and 8000 verbs. By applying rules to the dictionary, word list with more than 350000 words is generated.

Using this dictionary in compact form, significant improvement in web search results for Croatian language documents is achieved. Document index created with this technique is more consistent, and querying the database yields better results. Very good results are obtained in indexing newspaper and government sites. Our final aim is to create dictionary with about 50000 words marked by appropriate rules, covering most frequently used words of Croatian language and specialized fields like medicine, computer science, mathematics, architecture, etc. By loading different dictionary, this web search engine could be used for any other language which has *Ispell* dictionary files defined.

V. Conclusion

CONVENTIONAL search engines have no knowledge of language structure and syntax, which introduces drawback in performance of query matching. By incorporating language grammar into indexing process, we have moved step forward towards making process of extracting knowledge from the Internet more consistent. Benefits for the user are that more of the relevant documents are spotted. In our experiments we have shown that same query can return up to 60 times more results, depending on the language used in the documents and in the query. Great improvement is achieved for languages with complex set of grammar rules where each word can have many different forms. On the other side, dictionary contain-

ing all words from indexed documents is 5 to 10 times smaller, which reduces necessary computational power and memory capacities to perform both data indexing and query matching, thus lowering the cost of complete information retrieval system. By selecting *Ispell* dictionary format, this *IR* system can be used with number of languages with available dictionaries, and new languages can be added. As part of this project, Croatian dictionary with grammar rules is developed, and will be publicly available in its final form.

Acknowledgment: The authors would like to thank dr.sc. Marko Tadić from the Faculty of Philosophy of the University of Zagreb for his help and advice regarding computer implementation of Croatian grammar rules.

References

[1] Michael W. Berry, Zlatko Drmač, Elizabeth R. Jessup, *Matrices, Vector Spaces and Information Retrieval*,
[2] Michael W. Berry, Susan T. Dumais, Gavin W. O'Brien, *Using linear algebra for intelligent information retrieval*,
[3] Parry Housband, Horst Simon, Chris Ding, *On the Use of Singular Value Decomposition for Text Retrieval*,
[4] Demetris Zeinalipour-Yazti, Marios Dikaiakos, *High-Performance Crawling and Filtering in Java*,
[5] *Croatian National Corpus*, *http://www.hnk.ffzg.hr/cnc.htm*,
[6] Marko Tadić, *Natural Language Processing of Croatian and the Croatian National Corpus*, *Suvremena lingvistika, 41-43, 1996.*
[7] Gene H. Golub, Charles F. Van Loan, *Matrix Computations*,
[8] Google search engine, `http://www.google.com`.
[9] John W. Eaton, *GNU Octave*, 1997, `http://www.octave.org`.
[10] *Ispell, interactive spell-checking program* `http://www.gnu.org/software/ispell/ispell.html`
[11] Different articles from *http://java.sun.com/*