

Context Search

Pavol Navrat, Tomas Taraba
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
navrat@fiit.stuba.sk, tomastaraba@wms.sk

Abstract

Context search is related to gathering information about user's sphere of interest before search process. This information defines context and augments search query in subsequent phases of search to attain better search results.

There are several methods for searching with context. We took Rank-Biasing as a base and schemed out a modification hypothesizing that this will enable the method to fetch more relevant results in specific areas by taking advantage of knowledge based on additional information about that area.

In verification process we have done some experimental work and defined set of tests and metrics to compare the original and our improved version of the method. Experiments have shown that modified method attains better results for the same price.

1. Introduction

A person formulating search query on the web is doing this with some context in mind. In many cases he works with other documents, files or web pages. Information gathered from these documents can be used to find out the user's actual scope of interest. This information can be stored in some specific structure and then can be used to get better search results.

Respecting context in search can bring up more relevant search results. Many ways to augment search query by context can be explored. There are some differences in combining query and context together. Every method uses different algorithm to reach the goal, but usually they do the same work in different ways.

2. Related work

Context search methods are based on document's keywords acquisition and analysis. Keywords are gathered from documents recently viewed by user and stored in context. Context can be stored in *context term vector* [1].

Vector of terms contains keywords and the score of every keyword. The score is the number of keyword occurrences in all documents read by the user. Number of

context term vector dimensions equals the number of stored keywords.

After a context term vector is built, and user specifies the search query, this query is augmented by keywords from context. Methods for searching with context could be classified as [1,2]:

- *Query rewriting* – appends keywords from context to search query as a string and sends it to standard search engine.
- *Iterative-Filtering meta-Search* – generates many different sub-queries and sends them to multiple search engines. Results are re-ranked and aggregated into one set.
- *Rank-biasing* – Query and context of keywords are sent to modified search engine as complex query. At first, method finds documents matching query and then re-ranks them by fitness to context

Context
-roe:50
-egg:30
-hunter:10
-cookies:10

Figure 1. Context term vector

3. Improvement idea

Methods mentioned earlier are designed especially for respecting keywords from context. But is it enough? Isn't there any additional information that can be used to improve the search process and get more relevant results?

Consider user's age and level of ability to read and comprehend the text. There is a method to categorize readability of some text with the Flesch readability index [3]. "This method is used for estimating the reading comprehension level necessary to understand a written document. For a given document, the Flesch readability index is an integer indicating how difficult the document is to understand, with lower numbers indicating greater difficulty". For example Comics have readability index 95, New York Times 39, Auto Insurance 10 and American Tax Statute -6. Rudolf Flesch categorized readability indexes into 9 educational levels.

If we can take into mind, that user had never read a document classified as more difficult than a document comprehensible by high school student, is there a reason to include also documents understandable only by law school graduate?

There is also subject to put into context the name of the document's author. One author is often publishing

only documents related to small set of specific themes (politics, religion), or in reverse – one theme is written about by a small set of authors. Author’s name can also be a key to user’s sphere of interest.

Many document attributes can be found as significant in searching with context, so why not to include them in context and take them into account in the search and rank process?

Usually, different web pages do not have same type of attributes except keywords attribute. However, if we want to search with context, or do some recommendation based on context in some special area, analyzed methods cannot take advantage of knowing something more about the selected area. For illustration imagine an on-line library.

On-line library stores electronic publications. Every document has some metadata - attributes like number of pages, language genre, author, keywords, readability... Not all of them are eligible to include in search but there are some which can help to better identify user’s interest.

Consider user reading books in on-line library. He read two documents – genre: *fairy tale*, author: *Pavol Dobsinsky*, category: *traditional Slovak tales*, target age: *0 – 5 years old*. Out of these, the first was about *Little Red Riding Hood*, and the second about *The Wandering Egg*.

There are some concerns about the relevance of classic context search. Ensuring relevance in this case means to get results of documents containing Slovak fairy tales for children under 5 years instead of getting results of *Cookery-book for hunters*, which contains more keywords “*egg, roe, hunter*” than common Slovak fairy tale.

Our theory is, that respecting more document attributes than keywords can attain to ensure relevancy more than respecting only keywords. In the next parts of this text, we try to support this theory by designing and testing new method.

4. Context structure and acquisition

Context is additional information about user’s interest. We need to track more attributes than only that one for keywords. The way of context acquisition does not change, but the structure of stored data will be different.

On the beginning to do some changes in structure, we have to select which attributes to track and which not. In on-line library the set of tracked attributes can consist of the author, theme category, genre, and readability index. In job offers portal the context would be made of job location, minimum salary proposed and specialization.

Structure of the context will become more complex. Context will be represented by complex context vector, which is a vector of vectors. It will have number of dimensions equal to the number of document attributes used for search. Every dimension will represent one attribute and contains a ranked vector.

As an example: first dimension can be the dimension of keywords, next is dimension of authors and the last is the dimension of genres:

$$\vec{m} = \begin{pmatrix} \text{keywords} \mapsto (\text{roe} : 50, \text{egg} : 30, \text{hunter} : 10, \text{cookies} : 10) \\ \text{author} \mapsto (\text{pavol_dobsinsky} : 3, \text{maria_razusova} : 2) \\ \text{genre} \mapsto (\text{fairy_tales} : 3, \text{educational} : 1, \text{voyage} : 1) \end{pmatrix}$$

keywords	author	genre
-roe:50 -egg:30 -hunter:10 -cookies:10	-pavod_dobsinsky:3 -maria_razusova:2	-fairy_tales:3 -educational:1 -voyage:1

Figure 2. Modified context vector

We will acquire the context by tracking user’s clicks on our portal. Every time the user clicks to download some document, we analyze it and collect values of all tracked attributes. These values are added to context, or the score of value is only updated in case when the value is already stored in context.

As the number of documents viewed by user changes during the session, the context vector grows. In every search the actual context of that session is evaluated.

5. Search process

As the base for our approach we took Rank-Biasing method. We attempt to devise and incorporate some modifications to the original method to reach improvement idea.

5.1. Rank-biasing

The method has two arguments: *query* and *context*. Document search is made of two steps:

- Find documents which fit user query. For this step we can use standard search engine. Methods for ranking the results can be different in search engines, but we assume every score as a real number in range $\langle 1, 2 \rangle$, where the score of 2 is the best score and 1 is the worst score. The range $\langle 1, 2 \rangle$ is significant for the envisaged mathematical operations.

- Change the score of each result by the rate how much the result fits the context. The rate is the rank factor and is also a real number in range $\langle 1, 2 \rangle$. Factor 2 gets the document which absolutely fits the context (contains all terms from context) and factor 1 gets the document which does not fit context at all (contains no term from context) For calculating the rank factor we can use this formula:

$$F = 1 + \frac{N}{K}, \text{ where}$$

F is the rank factor of the new ranking,
 N is number of terms from context term vector which document contains
 K is total number of terms in context term vector.

new result score is the old score multiplied by rank factor
 $H(d) = H'(d) * F(d)$

5.2. Designed modifications

As we need to process complex context vector and include also other attributes in re-ranking, we modified the re-ranking function.

In the original method, only one factor (factor for keywords) was calculated. The modified method will calculate the number of factors equal to the number of complex context vector dimensions.

Consider one dimension by which the document will be ranked - z_x , where x is index of that dimension in complex context vector. Dimension z_x contains vector

$$\overline{z_x} = (h_1^x : r_1^x, h_2^x : r_2^x, \dots, h_N^x : r_N^x)$$

We define function $P(d, h, z_x)$. This function returns 1, if document d satisfies value $h \in \overline{z_x}$ in attribute represented by dimension z_x , otherwise returns 0.

The definition:

$$P(d, k, z_x) = \begin{cases} 1 & \text{if } h \in d[z_x] \\ 0 & \text{otherwise } k \notin d[z_x] \end{cases}$$

Rank factor for document d by dimension z_x can be calculated by this formula:

$$F(d, z_x) = 1 + \frac{\sum_{i=1}^N P(d, h_i, z_x) r_i^x}{\sum_i r_i^x}$$

Output of this function is real number from range $\langle 1, 2 \rangle$.

After documents are ranked by standard search engine, we must change the ranking by fitness to context.

Consider complex context vector containing K dimensions: (z_1, z_2, \dots, z_K) . Final new score $H(d)$ of document d is defined by formula;

$$H(d) = H'(d) * \prod_{i=1}^K F(d, z_x), \text{ where}$$

$H'(d)$ is original score assigned by standard search engine.

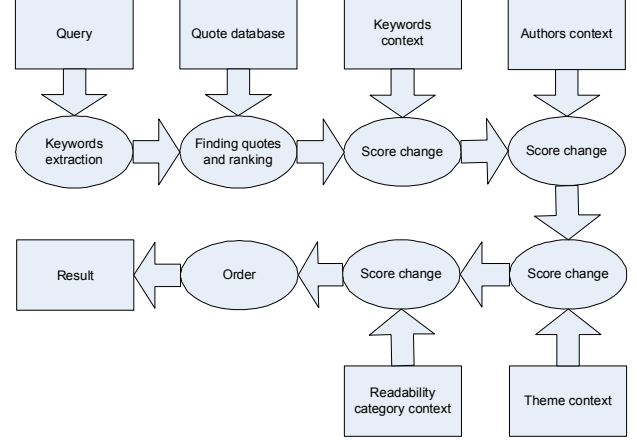


Figure 3. Rank-biasing algorithm modification

This way of impacting the score guarantees that if every found result does not fit the context at all (context has no effect) they will be ordered as if the context was not provided.

6. Testing

For ensuring that including other attributes in ranking has some effect, we compared original and modified rank-biasing method. Methods were tested in two aspects: the speed and the precision

As a specific area for testing we used database of quotes, where we tracked four attributes of each quote: keywords, theme category, author name and Flesch readability index. These attributes were used as dimensions of complex context vector.

6.1. Speed tests

The metric in this test was the response time – time in milliseconds needed to serve search request. We compared response time for original and modified method concerning the number of ranked results and the size of the context.

Measurement showed that speed of original and new method is nearly the same. The biggest delay into this process was still brought by ranking by keywords, which is the base of both methods, so response time value is nearly the same.

6.2. Precision and recall

This test is related to measure the quality of ranking function. We tracked the results given by new and original method and we compared them.

We tested both methods in several areas; the most challenging was this one: “a 13 years old user is searching

for quotes related to friendship or love and he or she reads only the simple-written ones, which understands better.”

In test we selected 500 quotes as the search result, re-ranked them and judged only first 30 in reordered set for the relevance. Context vector was size of 5, 15, 30 and 50 quotes from actually viewed and judged area.

Precision [4] (in %) was set by this formula:

$$precision = \frac{number_of_positive_results}{total_number_of_results}$$

As a positive results were classified that ones which belong to actually shot area. Total number of results was size of 30.

As the ranking by the theme category handicaps the original method in case when keywords were wrongly recognized, we made measurements with ranking by the theme category and without ranking by the theme category attribute.

For testing the resistance of registering bad documents into context we combined quotes related to actually pointed theme with quotes not related to actually pointed theme into context in ratio 1:5

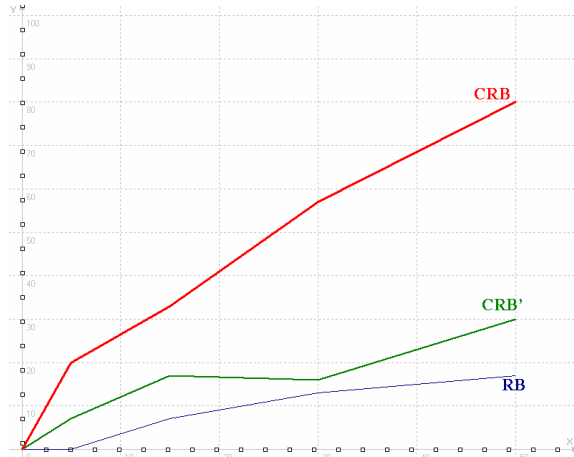


Figure 4. Precision, judging 30 results

On the chart in Figure 4 we can see a precision relative to the size of the context. There are 3 curves:

RB – results of Rank-Biasing method

CRB – results of new designed method

CRB' – results of new designed method without ranking regarding theme category.

Next standard metric for testing search engines is a *recall* [4]. Recall can be defined simply as the ability of a search engine to obtain all or most of the relevant documents in the collection.

As the rank-biasing method does not affect the number of found documents, and the purpose of this method is only to reorder search results to move better to the top, there is no subject to measure recall in our experiment.

7. Summary

We used rank-biasing as a base for our experiments, as an example how to design changes. But there was no other reason for using it than rank-biasing is simple. It provides good and transparent environment for demonstrating improvement idea and changes we wanted to design.

What we changed in the original method was the ranking function. The way how we designed the modification is useable in other approaches as well.

For an example of particular use of this ranking function could be taken a method called “*Web search engine as a bee hive*.” [5] This is the approach to web search that is based on the bee hive metaphor. A model of searching documents on the web is presented as an analogy to searching for a best quality food by a bee. This model is also a good environment for examining our improvement idea, while the modification can be easily incorporated as a modification of bee’s view on food quality. A bee can take a measure of found document quality respecting context in way we defined in design.

After some other modifications, this method can be used for recommender systems on the web, where context can be used to find out what is user interested in and recommend him the document (e-book, product, job offer...)

8. Acknowledgements

This work was partially supported by the Slovak State Programme of Research and Development “Establishing of Information Society” under the contract No. 1025/04 and the Slovak Research and Development Agency, grant No. APVT-20-007104.

9. References

- [1] R.Kraft, C.C.Chang, F.Maghoul, R.Kruman. Contextual Search Methods In Context Search. Yahoo! Inc, USA, 2006
- [2] R. Kraft, *Cost-effective creation of specialized search engines*, Santa Cruz, 2005
- [3] Kincaid J.P., Fish R.P., Rogers R.L., Chissom B.S.; Derivation of new readability formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula), 1975
- [4] Clarke, S., & Willett, P. Estimating the recall performance of search engines. ASLIB Proceedings, 1997
- [5] P. Navrat, M. Kovacik. Web Search Engine as a Bee Hive. In: 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2006 Main Conference Proceedings)(WI'06), Hong Kong, 2006, pp. 694-701.