Personalized News Search in WWW: Adapting on user's behavior

Christos Bouras, Professor
Research Academic Computer Technology Institute,
Rion, Patras, Greece, GR26500
bouras@cti.gr

Vassilis Poulopoulos, MsC
Research Academic Computer Technology Institute,
Rion, Patras, Greece, GR26500
poulop@cti.gr

Panagiotis Silintziris
Computer Engineer and Informatics Department
Rion, Patras, Greece, GR26500
silingir@ceid.upatras.gr

Abstract— **Personalized Web Search becomes nowadays a promising option in the field of Information Retrieval and search engines design by improving both output quality and user experience. In this paper, we present and evaluate the subsystem, which conducts the Advanced and Personalized search of PeRSSonal, a web-based mechanism for the retrieval, processing and presentation of articles and RSS feeds collected from major news portals of the Internet. The proposed technique uses information explicitly provided by the user in his profile as well as information that the mechanism can learn from the user's behavior during his search and browsing sessions in the system. As this behavior dynamically evolves, the same happens to the user's interests under the prism of the search engine. By adopting this user-centric approach, we manage to present the user with better-refined and more focused results, incorporating his personal preferences to the output. The algorithm operates not in a stand-alone manner but it co-operates and binds with the rest of modules of PeRSSonal in order to accomplish maximum integration with the system. Furthermore, we introduce an enhancement in the search function, based on cached results from past search sessions of each user individually.**

Keywords-Article search engine, personalized search, peRSSonal.

I. INTRODUCTION

The technological advances in the World Wide Web, the low cost and the ease of access to it from any place in the world by using, not only conventional computers but also portable devices and cellular phones with advanced networking capabilities, has dramatically changed the way people face the need for information retrieval. More and more users migrate from traditional mass media to more interactive digital solutions such as Internet news portals. These digital neighborhoods provide a direct link to fresh and unfiltered stream of data, giving their registered users the opportunity to stay up to date, in real time, with news and all kinds of information regarding their personal interests from all over the globe. This increasing popularity of Internet, as a vast digital data pool, which grows in an exponential rate, combined with the rather static and unchanged nature of human vocabularies, does not come without problems: Over time, it becomes a tedious task for the average user to successfully select a proper set of keywords that best describe his question and then locate the "right" piece of information in an ocean of irrelevant data.

Under these circumstances, search engines are deployed on most news portals to help users find what they are looking for with less effort. In the majority of these search engines, when different users submit the same query, the same results are returned, in the same order, regardless of who submits the query. Obviously, it is unlikely that all the users of a search engine are so similar in their demands that a sole approach to searching fits all needs. Indeed, in terms of searching, one half of all retrieved documents have been reported to be irrelevant with what the user expected [1]. Additionally, a number of studies have shown that a vast majority of queries to search engines are short and underspecified [2] and different users may have completely different intentions for the same query [3]. The explanation is simple as one keyword or a limited set of keywords cannot always be an unambiguous guide to determine what a user is exactly interested in. This is the point where the personalized search can be of essential help. Presumably, information retrieval will be more efficient if individual users' idiosyncrasies are taken into account. Such a search strategy could decide autonomously for each user whether he is interested in an article and, in the opposite case, prevent it from being displayed. By modeling the user appropriately and personalize search according to his individual demands, we can achieve an improvement in the retrieval accuracy.

Throughout literature, we can identify two major directions for search personalization. Query expansion and Results processing can complement each other and by understanding both the user and the context, a breakthrough in search efficiency can be achieved [4]. Query expansion is a way of solving the problem of word mismatch, which arises when users employ different terms than those used by content authors, to describe the same concept. This is done by augmenting queries length with more relative words or phrases and it may offer a efficient solution [5]. Context can take different forms, like category manually selected by the user [6] or combination of titles and descriptions of clicked search results after an initial query has been submitted [7]. On the other hand, result processing includes filtering and reorganizing of the search results in order to provide the user with a more refined output. This filtering can be either in the domain of the returned results [8], eliminating in this way documents irrelevant to specific web domains, or in the news items that may not be of interest to a given user, according to that user's explicit (through rankings) or implicit (viewing and order duration) feedback, leading in personalized views of the result. Another approach to result processing deals with reorganization and re-ordering of the results, which is one of the major ideas of our work. This may involve the construction of a user profile over time with resources such

as issued queries and visited links as in [9]. Previous queries and summaries of clicked results could also be used for re-ranking in the current session as in [10]. In addition to these server-side techniques, some client-side techniques have been proposed in the past, as in [11], where query expansion and result re-ranking are employed on the basis of the immediately preceding query and of summaries of viewed results.

In our work, we use an efficient combination of query expansion and results processing to produce personalized output. The difference with the aforementioned implementations, regarding the query expansion, is that it takes place after the first unranked set of result has been retrieved, participating in this way in the result processing. In the procedure which we will describe, we expand the user's query with keywords that the engine has selected for a category or for a user. This selection is based on the user's previous search sessions and the categorization system of PeRSSonal [14]. By assigning increased or decreased importance weights to some keywords, it becomes feasible for the engine to obtain some kind of knowledge over the user's preferences. Thus, for each resulting article, a relevance factor is computed from keywords weight allowing the re-ranking of articles according to it. The remaining of the paper is structured in the following manner: section 2 describes the architecture of the system with the focus in the core of the personalized search algorithm, which is analyzed in Section 3. In section 4, we present some experimental results and evaluation of our work and we conclude in Section 5 with some remarks about the described techniques and future work..

II. ARCHITECTURE

The architecture of the system is distributed and based on standalone subsystems but the procedure to reach at the desired result is actually sequential, meaning by this that the data flow is representative of the subsystems of which the mechanism consists. This section is a description of how these features are integrated into the mechanism. We are putting the focus on the personalized search subsystem, though brief analysis of the other modules is presented in order to cross-connect the features of our system.

A. Components

The architectural schema consists of a series of subsystems, as depicted in Figure 1. The collaboration between the distributed parts relies on the open standards for input and output that are supported by each part of the system and on the communication with a centralized database. The general procedure is as follows: at first, web pages are captured and only the useful text (drop stop words, punctuation etc) is extracted from them. Then, the extracted text is parsed followed by summarization and categorization. Finally we have the presentation of the personalized results to the end user.

For the first step, a simple web crawler is deployed, which uses as input the addresses extracted from the RSS feeds. Theses feeds contain the web links to the sites where the articles exist. The crawler fetches only the html page, without elements such as referenced images, videos, css or JavaScript files. Thus, the database is filled with pages ready for input to the 1st level of analysis, during which, the system isolates the "useful" text from the html source. Useful text contains the article's title and main body. In the 2nd level of analysis, XML files containing the title and the body of articles are received as input, targeting at applying pre-processing algorithms on this text in order to provide as output the keywords, their location in the text together with their absolute frequency in it (number of times met in the text). These results are the primary input to the 3rd level of analysis. In the 3rd level of analysis, the summarization and categorization technique takes place. Its main scope is to characterize the articles with a label (category) and come up with a summary of them. Details about these procedures can be found in [12, 13, 14]. The core of our work can be found in the 4th level of analysis, where the results are presented to the end user in a personalized view. The algorithms of personalization, which will be analyzed in the next section, take as input, information about the user's profile and preferences, collected and processed during his past sessions in the portal. For each user, a set of keywords with assigned relevance weights is used as the main criteria for the final order in which the articles will be presented as well any extra articles that the engine considers as "possibly interesting" for the specific user, although these articles might not be directly linked to the specified query.

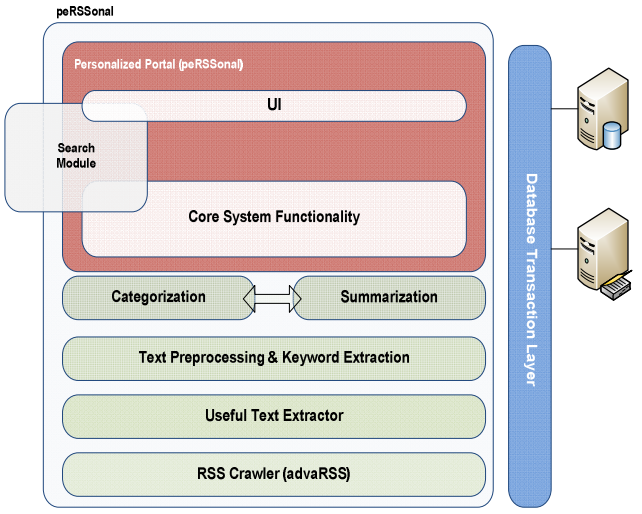


Figure 1: The search module within peRSSonal architecture

B. Flow of Information

In Figure 2, we can see the general schema and flow of the advanced and personalized search sub-module. The user submits the keywords together with the search configuration options through a form. The keywords first pass through a Stemmer so that they get in the form that they are stored in the database. A caching algorithm is used to search for similar queries submitted in the past from the same user, and if matches are found, cached results are directly obtained improving in this way the search speed. In the next step, the query is enriched with more keywords, relative to the user's

profile acting as a base for more relevant articles to be retrieved. In the final step, a weighting algorithm is deployed to adjust the weights of the keywords that the engine has characterized as favorite for the user who conducts the query, according to his past searches and general behavior in the system.

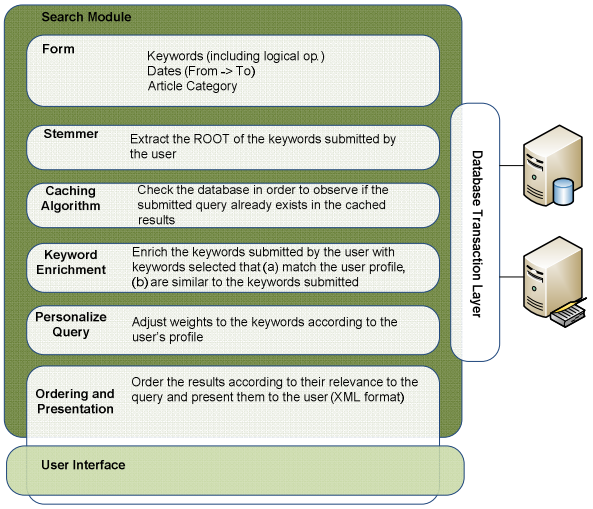


Figure 2: Search Module Operation

III. ALGORITHMIC ASPECTS

A. *Configuration and Keyword Refinement*

Before the engine triggers the search procedure, the user has first to configure the search. Apart from the specified keywords, a few other options are offered, including the date period for the target result, the selection of the logical operation ("OR" and "AND) which will be performed in the articles matching and the thematic category of the desired output. Before proceeding with query search operation, the engine passes the keywords through a stemmer which implements the Porter Stemming Algorithm on the English language. Thus, we enable the integration of the search engine with the rest of system, which is build on stems rather than full words for the articles categorization. Additionally, simple duplicates elimination is executed on the stems.

B. *Fetching of Results*

In the first phase of the algorithm, we fetch from the database the articles that constitute a direct match to the submitted query, taking into account the configuration of the search options. We should emphasize here, that if the user has requested articles of a particular category, then an article is fetched only if it has the biggest frequency in the specified category among all categories. This can be straightforwardly achieved, as PeRSSonal maintains the frequency per category for each article it collects from the Internet, by associating the frequency of its keywords with every thematic category. The results are temporarily stored in array structures so that they can be sorted in later stages.

C. *Enrichment and Personalization*

In the next phase of the search procedure, we have to refine the order of the articles to be shown both for the case of a generic and for the case of a personalized search.

In the first case of a generic search of an anonymous user, the query is expanded with more keywords so that results with highest relevance to the search request can be obtained. This algorithm takes each keyword in the query and assigns a weight to it. Keyword weights in our experiments start from 0.1 for the first keyword in the query, with each next keyword having weight 0.01 less than the previous one. We based this choice on the assumption that in a query string, the user writes the most important keywords at the beginning. However, a different pattern could also be an option. Furthermore, for each keyword, we compute its absolute frequency for all categories of PeRSSonal. The purpose is to expand the query of an anonymous user by enriching it with more keywords from categories with high relevance to the query keywords. To accomplish it, we use in our experiment different representativity (relativity) factors ranging from 1.5 to 4.5. The way this factor affects the number of the extra added keywords is described in section 4. For our system, a representativity factor R means that a keyword K is considered as representative of category C, only if its absolute frequency of appearance in C is R times higher compared to its absolute frequency in the category in which it has its second highest absolute frequency. Consequently, higher values of R yield lower possibility for the user to select category-representative keywords and for the engine to enrich the query, while lower values allow for easier query expansion with more keywords to refine the results. In case we find high relevance for a category, we proceed with enriching the query with more keywords from this category. For this reason, we retrieve the keywords having absolute frequency greater or equal than the particular keyword's frequency and we select these two that have frequency just above or equal to the query's keyword. This query may produce several new keywords, which we call child keywords. Our rule for child-weighting is that each new child takes the weight of its parent (for keywords with more than two parents in the same query, the higher weighted parent is chosen).

After the enrichment process has been completed and the weights of all keywords have been adjusted, including the keywords added to enrich the query, we can compute the relevance of an article to the users query according to the following general formula:

$$I_{relevance} = \frac{\sum_{i=1}^{N}(x_i y_i)}{\sqrt{\sum_{i=1}^{N}(x_i^2)}\sqrt{\sum_{i=1}^{M}(z_i^2)}}$$

In this formula, x_i , y_i denote respectively the computed weight and the actual frequency of the ith keyword in the expanded query, which consists of N keywords and z_i denotes the actual frequency of the ith keyword of the list of all M keywords existing in the fetched article. Obviously,

this index gives a normalized measure of an article's relevance to a given query.

In the case of a personalized search, the relevance of each article to the query has to be computed by taking into account the individual profile and the preferences of the user committing the search request. For the profile of each user, during the registration process we assign different weights to different groups of keywords in order to better describe his explicit preferences. The logic behind the variation in the keywords weights is that keywords with high relevance over a user's favorite thematic category gain positive weights, while keywords belonging in categories which are of less or of no interest to the user have lower or negative weights accordingly. The profile for each user is initialized during his registration into the system, where he can provide an index of interest (-5 to 5) for each thematic category.

This profile can change dynamically overtime in accordance to the user's overall behavior inside the system (visited articles, time spent over an article etc.). Based on such profiles, we can compute and index of relevance of each article for any given user. The formula used for this reason in our implementation of the personalized search is the following:

$$I_{personalized} = \frac{\sum_{i=1}^{N}(p_i w_i)}{\sqrt{\sum_{i=1}^{N}(p_i^2)}\sqrt{\sum_{i=1}^{M}(z_i^2)}}$$

where N is the number of keywords that have been assigned a weight (negative or positive) in the user's profile, p_i is the weight of each such keyword, w_i is the actual frequency of each such keyword in the particular article, M is the total number of keywords in the article and z_i are their actual frequencies inside the article. From this formula we can see how highly weighted keywords in the user's profile can increase the relevance of an article, therefore the rank in which it will be finally displayed and that for different users we obtain different relevance for a given article. For example, upon registration process, if a user has declared high interest over education and no interest over sports, then keywords such as "school", "pupil", "teacher", etc have obtained high weight leading in high computed relevance for articles in the education category, while keywords such as "football", "athlete", etc. have negative weights, leading in low relevance for articles around sports. Finally, by sorting, the fetched articles array according to their computed relevance, we manage to present the user with results personalized and targeted to his profile and preferences.

However, in order to give our technique an ability of constant-on-user personalization, these weights (initialized upon registration) do not remain static for a user throughout all his next search sessions inside the portal and in this point we can see the dynamic nature of our algorithm. His profile dynamically evolves overtime in accordance to his overall "behavior" inside the system (thematic category of visited articles, time spent on an article, articles ignored, etc.). We constantly update weights of keywords in his profile every time he executes a query. If, for example, the user starts to use systematically some keywords, which previously had low or negative weight due to never or rarely been used, their weight will begin to increase. In the opposite direction, keywords with high weight, which are no more or rarely being used, will start to reduce in weight. Thus, it is possible to better refine the future search results of this user and keep the thematic orientation of the output as close as possible to the evolution of the his main interests.

D. *Caching on Results*

Prior to the results fetching, another action is taken, by which we aim at improving the speed of our search engine. We introduced in our algorithm a method based on caching, that, when applied to a user's consecutive or near search sessions inside the portal, it can achieve reduced speed, allowing for better overall user experience, while simultaneously reducing the workload of the system. Our caching algorithm, stores in the database the options, the keywords and the results for each query. This happens for each query in real-time (when executed). In the future searches, over a short period of time, we can use the cached results as a fast and system relieving response to the user whenever he submits a similar query matching the stored one.

IV. EXPERIMENTAL EVALUATION

A. *Representativity Factor*

In the query expansion process, we manage to enrich the user's query by using a number that we call representativity factor. This number represents the possibility the user has in selecting keywords that are highly representative of some categories. When the user picks these keywords in his query, the engine expands the query by enriching it with more keywords from the category with the highest relevance. In Figure 3, we present the relation between the representativity factor and the number of keywords which automatically become category representative. This diagram was constructed with experimental results over our database. We first obtained the 1000 most representative keywords for 7 of the basic thematic categories existing in the articles database. We removed the duplicates and we ended with 490 different keywords. As we can see from this graphic, the possibility of selecting category representative keywords is relatively high, almost 70%, when the representativity factor takes values below 2 and falls at 50% and lower for values above 3.5, making it more "difficult" to select representative keywords. The extra added keywords do not participate in the creation of the articles set, which will be presented to the user, but their role is to aid in the re-ranking procedure of the final results. We should notice that neither low values for the factor should be used, as this would make it very possible to reorder the output articles based on keywords the user had not in mind when submitting the query, nor low values as this would make the final articles result difficult to focalize one the thematic category, but it would spread among different categories. For the experiment of the next

paragraph, a representativity factor of 3 was used in the algorithm.

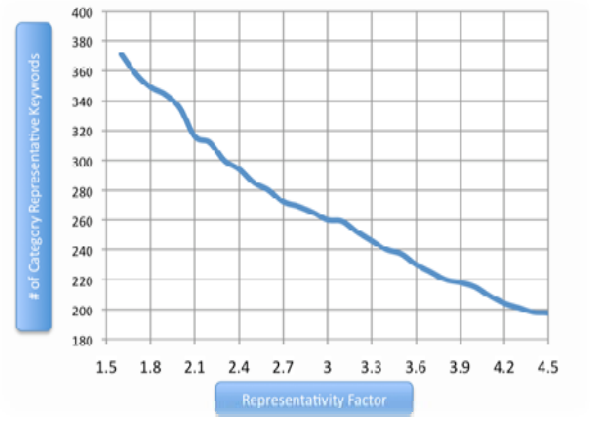


Figure 3: Representativity Factor

B. Personalized VS Generic Search

In order to compare the results of a personalized search to the results of a non personalized search, we conducted our experiments for several virtual users of the portal. During the registration process, each user provided a positive preference bias over one or two categories and negative bias for all other categories of personal, simulating in this way different groups of people. This led into a different initial profile for each user with increased weights on different subsets of keywords, as described in paragraph 3.3. In the charts at the end of the paragraph, we consider a single user A as a user with high preference over sport news to present the experimental results. This said, a user with favor over a particular category, is not excluded from being presented or selecting to view articles from different categories but the engine tries to adapt as much as possible the results to his profile. The queries of the experiment consist of category independent keywords so that the results we obtain are generic. Examples of such keywords are 'Sunday', 'New York', 'red', 'environment' etc. In this point we should notice that prior to gathering the presented results, we trained the system for each user separately with articles that were of high interest for these users. As a result, his profile became more category-polarized on the category the user initially indicated as categories of interest.

In the first phase of the experiment, we conducted a non personalized search (representing an anonymous user) on a generic query. We take measurements for the first 200 articles returned from the search, although obviously the vast majority of users would never reach to browse so many articles. As expected, due to the generic character of the query, the result set of articles we come up with consisted of articles spread among all categories of our database. In this way we guarantee that all users, independently of their favorite category, can find interesting articles in the results set. For each article, we kept its rank in the results page, the category in which it belongs (most representative category) and its relevance to the query in order to compare these data with the corresponding results of the personalized search of the next phase. We had each virtual user to select and browse 15 of these articles, mostly consisting of articles on his favorite category, for which we also kept track of their ids and ranks.

In the second phase of the experiment, we conduct a personalized search for each one of the users, executing the same query as in the generic search. This time, we keep track for each user the first 50 articles of the search result returned. For the evaluation of the output, we compared for the selected articles by each user the rank in which they were fetched in the generic search of the first phase with the rank that they are fetched in this phase. Most of the articles that the user selected in the first phase came up in much higher ranks in this phase. Also their relevance increased as it is computed based on the profile and keywords weights. For users with one favorite category, selected upon registration, this situation was even more obvious, because of their strong focus over one specific category.

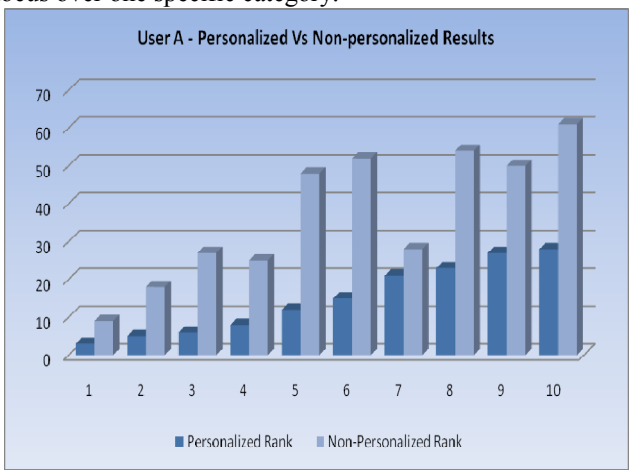
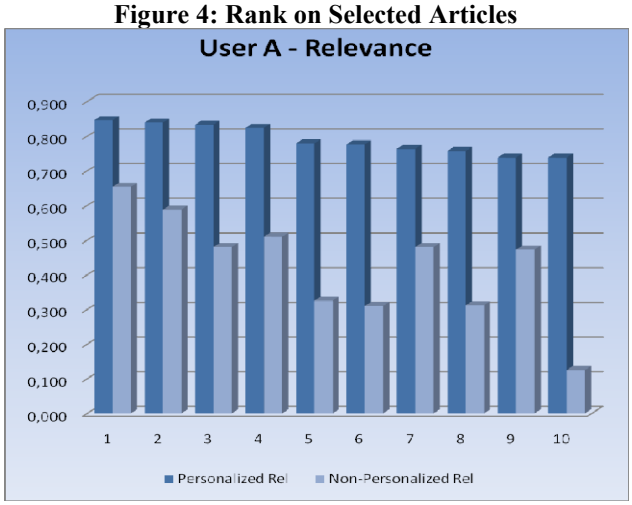


Figure 4: Rank on Selected Articles



For example, as we can see in Figure 4 for user A, 10 out of the 15 articles he had selected in the non personalized search within the first 200 results have moved several places higher in the rank of the personalized search, while 5 of them have climb up in the first 10 result articles. One significant point is that, overtime, for users that are consistent on browsing articles of one category, the results can be even more biased as keywords of categories out of interest get

very negative weights. In figure 5, we can also see the relevance with which these 10 articles were returned both in generic and personalized search. This chart shows that in the case of a personalized search the user sees articles with a lot higher relevance on his query in the beginning of the output articles set.

In the case of an anonymous user our algorithm simulates the personalized search procedure. The engine can select to present him with results as if he was conducting a personalized search by trying to match the submitted query with a specific category, whenever it is possible, and then execute the query as if this user had a profile with preference over this specific category. Of course, in this case we cannot learn from the user's profile or adapt to changes as no profile exists, but we can still benefit as the results are more focalized on the query's thematic orientation and there is high probability that the user is more satisfied with the search output.

V. Conclusion and Future Work

Due to the dynamism of the Web, the content of the web pages change rapidly, especially when discussing about a mechanism that fetches more than 2500 articles on a daily basis and presents them personalized back to the end user. Personalized portals offer the opportunity for focalized results though, it is crucial to create accurate user profiles. Based on the profiles that are created from the peRSSonal mechanism we created a personalized search engine for peRSSonal web portal system in order to enhance the searching procedure of the registered and the non-registered users. Focalizing on the registered users, meaning that we own information about their preferences, we presented a system that is able to personalize the search results on each user's profile and keep the personalization procedure consistent independently of the changes that the profile may undergo. We presented the algorithms and the formulas that lead to personalizing the results and more specifically the ordering of results in order to push the relevant articles to the top of the results for a specific user's profile.

Comparing the results to the generic search's results it is obvious that the system is able to enhance the searching procedure and help the users locate the desired results more easily. The system, as every web based system, exports its results on XML format in order to assure a universal output format. For the future what we would like to do is to further enhance the whole system with a more accurate search personalization algorithm in order to make the whole procedure faster and in order to omit any results that are of very low user interest.

Furthermore, what is to be done is the creation of a caching subsystem in order to have the results of the user's queries stored and thus the results could be presented faster. Finally, a user grouping mechanism can be added to the personalized search system in order to propose some articles on same or similar queries that were selected from users with similar profile.

References

[1] E. Casasola. Pro Fusion Personal Assistant: an agent for personalized information filtering on the WWW. Master's Thesis, The University of Kansas, Lawrence, KS, 1998.

[2] B.J. Jansen, A. Spink, and T. Saracevic. Real life, real users, and real needs: A study and analysis of user queries on the Web. Information Processing and Management, , 2000, 36(2): pp. 207 – 227

[3] Robert Krovetz and W. Bruce Croft. Lexical ambiguity and information retrieval. Information Systems, 1992, 10(2): pp. 115-141

[4] S. Lawrence, Context in Web Search. IEEE Data Engineering Bulletin. 2000, 23: pp. 25 – 32

[5] J. Xu and W.B. Croft. Query Expansion using local and global document analysis. In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Zurich, Switzerland, 1996, pp. 4-11

[6] Glover, E., Lawrence, s., Brimingham, W.,Andgiles, c. L. Architecture of a meta search engine that supports user information needs. In Proceedings of the 8th International Conference on Information Knowledge Management. Kansas City, MO, 1999, pp. 210–216.

[7] Leory, G., Lally, a. M.,Andchen. The use of dynamic contexts to improve casual internetsearching.ACM Trans. Inform. Syst. 2003, 21(3): pp. 229–253.

[8] Oyama, S., Kokubo, T.,Andishida, T. Domain-specific Web search with keyword spices..IEEE Transformation Knowledge and Data Engineering. 2004, 16(1): pp. 17–27.

[9] Teevan, J., Dumais, S. T.,Andhorvitz, E. Personalizing search via automated analysis of interests and activities. In Proceedings of 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Salvador, Brazil, 2005, pp. 449–456.

[10] Shen, X., Tan, B.,Andzhai, C. X. Context-sensitive information retrieval using implicit feedback. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Salvador, Brazil, 2005, pp. 43–50.

[11] Shen, X., Tan, B.,Andzhai, C. X. Implicit user modeling for personalized search. In Proceedings of the 14th ACM International Conference on Information and Knowledge Management. Bremen, Germany, 2005, pp. 824–831.

[12] C. Bouras, V. Poulopoulos, V. Tsogkas. Efficient Summarization Based On Categorized Keywords. The 2007 International Conference on Data Mining (DMIN07), Las Vegas, Nevada, USA. 2007, pp. 25 - 28.

[13] C. Bouras, C. Dimitriou, V. Poulopoulos, V. Tsogkas, The importance of the difference in text types to keyword extraction: Evaluating a mechanism, in: International Conference on Internet Computing, CSREA Press, 2006, pp. 43–49.

[14] C. Bouras, V. Poulopoulos, V. Tsogkas. PeRSSonal's core functionality evaluation: Enhancing text labeling through personalized summaries. Data and Knowledge Engineering Journal, Elsevier Science, 2008, 64(1): pp. 330 – 345. (FLAG)