Sailing the Web with *Captain Nemo*: a Personalized Metasearch Engine

Stefanos Souldatos STEF@DBLAB.ECE.NTUA.GR
Theodore Dalamagas DALAMAG@DBLAB.ECE.NTUA.GR
Timos Sellis TIMOS@DBLAB.ECE.NTUA.GR
School of Electrical and Computer Engineering, National Technical University of Athens, Athens, GR, 157 73

Abstract

Personalization on the Web is an issue that has gained a lot of interest lately. Web sites have already started providing services such as preferences for the interface, the layout and the functionality of the applications. Personalization services have also been introduced in Web search and metasearch engines, i.e. tools that retrieve Web pages relevant to keywords given by the users. However, those services deal mostly with the presentation style and ignore issues like the retrieval model, the ranking algorithm and topic preferences. In this paper, we present *Captain Nemo*, a fully-functionable metasearch engine that exploits personal user search spaces. Users can define their personal retrieval model and presentation style. They can also define topics of interest. *Captain Nemo* exploits several popular Web search engines to retrieve Web pages relevant to keywords given by the users. The resulting pages are presented according to the defined presentation style and retrieval model. For every page, *Captain Nemo* can recommend a relevant topic of interest to classify the page, exploiting nearest-neighbour classification techniques.

1. Introduction

Nowadays, huge volumes of data are available on the Web. Searching for information is extremely difficult, due to the large number of information sources and their diversity in organizing data. Users should not only identify these sources, but also determine those containing the most relevant information to satisfy their information need.

Search and metasearch engines are tools that help the user identify such relevant information. Search engines retrieve Web pages that contain information relevant to a specific subject described with a set of keywords given by the user. Metasearch engines work at a higher level. They retrieve Web pages relevant to a set of keywords, exploiting other already existing search engines.

Personalization on the Web is an issue that has gained a lot of interest lately. Web sites have already started providing services such as preferences for the interface, the layout and the functionality of the applications. Personalization services have also been introduced in Web search and metasearch engines. However, those services deal mostly with the presentation style and ignore issues like the retrieval model, the ranking algorithm and topic preferences.

In this paper, we present *Captain Nemo*, a fully-functionable metasearch engine that creates personal user search spaces. Users can define their personal retrieval model. For example, they can select the search engines to be used and their weight for the ranking of the retrieved pages, the number of pages retrieved by each engine, etc. Users can also define topics of interest. For every retrieved Web page, *Captain Nemo* can recommend a relevant topic of interest to classify the page, exploiting nearest-neighbour classification techniques. The presentation style is also customizable, as far as the grouping and the appearance of the retrieved pages is concerned.

A typical application scenario for *Captain Nemo* starts with a set of keywords given by the user. *Captain Nemo* exploits several popular Web search engines to retrieve Web pages relevant to those keywords. The resulting pages are presented according to the user-defined presentation style and retrieval model. We

Appearing in *W4: Learning in Web Search, at the 22nd International Conference on Machine Learning*, Bonn, Germany, 2005.

note that users can maintain more than one different *profiles* which result to different presentation styles and retrieval models. For every retrieved Web page, *Captain Nemo* can recommend relevant topics of interest to classify the retrieved pages, exploiting nearest-neighbour classification techniques. User can optionally save the retrieved pages to certain folders that correspond to topics of interest for future use.

Contribution. The main contributions of our work are:

- We present personalization techniques for metasearch engines. These techniques do not only deal with the presentation style but also with the retrieval model and the ranking of the retrieved pages.
- We suggest semi-automatic classification techniques in order to recommend relevant topics of interest to classify the retrieved Web pages.
- We present a fully-functionable metasearch engine, called *Captain Nemo*[1], that implements the above framework.

Related Work. The need for Web information personalization has been discussed in (Shahabi & Chen, 2003; Sahami et al., 2004). Following this, several Web search and metasearch engines[2] offer personalization services. For example, Alltheweb offers the option to use personal stylesheets to customize the look and feel of its search page. Altavista provides styles to present the retrieved Web pages with high or low detail. The metasearch engines WebCrawler, MetaCrawler, Dogpile can group the Web pages according to the search engine that actually retrieves them. Regarding the retrieval model, several metasearch engines let the user define the search engines to be used (e.g. Query Server, Profusion, Infogrid, Mamma, Search, Ixquick). Some of them (e.g. Query Server, Profusion, Infogrid, Mamma) have a timeout option (i.e. time to wait for Web pages to be retrieved). Also, Query Server and Profusion offer the option of setting the number of Web pages retrieved by each engine. To the best of our knowledge, there is not any metasearch engine that offers the option of setting the weights of the search engines for the ranking of the retrieved pages.

Concerning the topics of interest, Buntine et al. (2004) claim that topic-based search will be necessary for the next generation of information retrieval tools. The search engine Northern Light[3] has an approach called *custom folders* that organizes search results into categories. Inquirus2 (Glover et al., 2001) uses a classifier to recognize web pages of a specific category and learn modifications to queries that bias results toward documents in that category. Chakrabarti et al. (1998) proposes statistical models for hypertext categorization by exploiting link information in a small neibourhood around documents.

Outline. The rest of this paper is organized as follows. The personalization features of *Captain Nemo* are discussed in Section 2. Section 3 presents the classification algorithm that recommends relevant topics of interest to classify retrieved Web pages. The architecture of *Captain Nemo* and several implementation issues are discussed in Section 4. Finally, Section 5 concludes this paper.

2. Maintenance of User Profiles

Captain Nemo maintains user profiles for different presentation styles and retrieval models. A user can have more than one different *profiles* which result to different presentation styles and retrieval models. Figure 1 illustrates the personal search space offered to users by *Captain Nemo*. We next discuss the available personalization options for the retrieval model, the presentation style and the topics of interest.

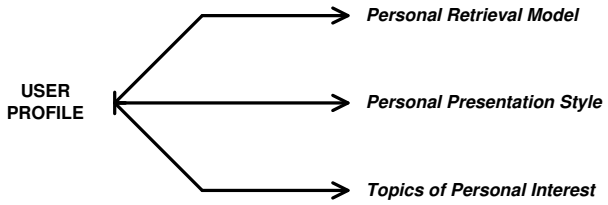

Figure 1. Personal search space offered by *Captain Nemo*.

2.1. Retrieval Model

As seen before, most of the existing metasearch engines employ a standard retrieval model. In *Captain Nemo*, this restriction is eliminated and users can create their own retrieval model, by setting certain parameters in the system. These parameters are described below:

Participating Search Engines. Users can declare the search engines they trust, so that only these en-

[1] http://www.dbnet.ece.ntua.gr/~stef/nemo/

[2] Google, Alltheweb, Yahoo, AltaVista, WebCrawler, MetaCrawler, Dogpile, etc.

[3] http://www.northernlight.com/index.html

gines are used by the metasearch engine.

Search Engine Weights. In a metasearch engine, retrieved Web pages may be ranked according to their ranking in every individual search engine that is exploited. In *Captain Nemo* (as shown in Section 4), the search engines can participate in the ranking algorithm with different weights. For example, a lower weight for a search engine indicates low reliability and importance for that particular engine. Users have the option to set their own weights for every search engine exploited by *Captain Nemo*.

Number of Results. A recent research (iProspect, 2004) has shown that the majority of search engine users (81.7%) rarely read beyond the third page of search results. Users can define the number of retrieved Web pages per search engine.

Search Engine Timeout. Delays in the retrieval task of a search engine can dramatically deteriorate the response time of any metasearch engine that exploits the particular search engine. In *Captain Nemo*, users can set a timeout option, i.e. time to wait for Web pages to be retrieved for each search engine. Results from delaying search engines are ingored.

2.2. Presentation Style.

Users of *Captain Nemo* can customize the look and feel for the presentation of the retrieved Web pages, having the following options:

Grouping. In a typical metasearch engine, the results returned by search engines are merged, ranked and presented as a single list. Beside this typical presentation style, *Captain Nemo* can group the retrieved Web pages (a) by search engine or (b) topic of interest pre-defined by the user. The latter is based on a semi-automatic classification technique which will be described in Sections 4. Figure 2 illustrates an example where retrieved Web pages are grouped by topic of interest.

Content. The results retrieved by *Captain Nemo* include the page title, page description and page URL. The user can declare which of these parts should be displayed.

Look and Feel. Users can customize the general look and feel of the applications. They can select among color themes and page layouts to define different ways of presenting results. Figure 3 shows the available options for customizing the look and feel of the application.

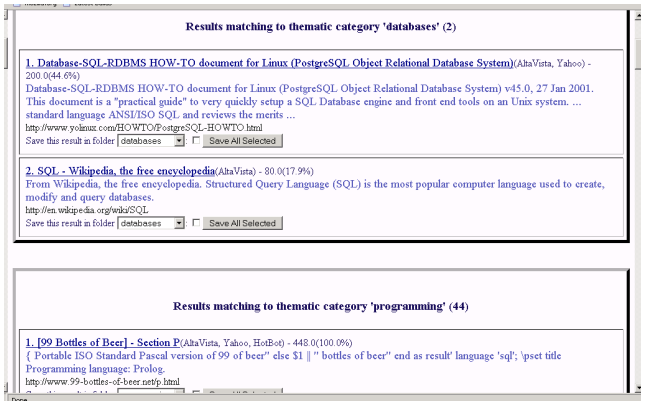

Figure 2. Grouping of retrieved Web pages by topic of interest.

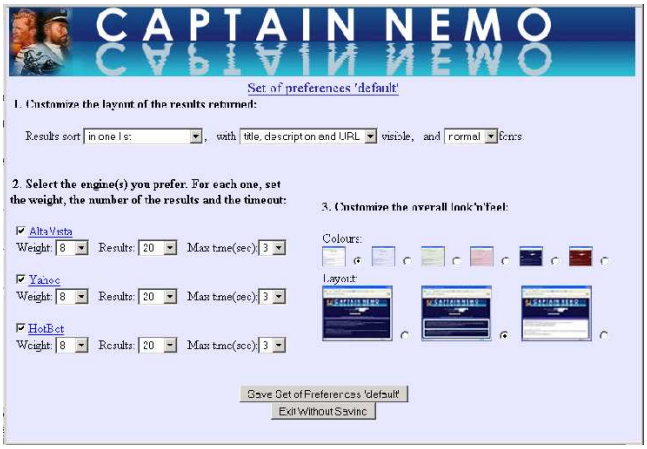

Figure 3. Editing set of preferences.

2.3. Topics of Interest

In *Captain Nemo*, the retrieved Web pages are presented according to the user-defined presentation style and retrieval model. For every retrieved Web page, *Captain Nemo* can recommend relevant topics of interest to classify the retrieved pages. Users can optionally save the retrieved pages to certain folders that correspond to topics of interest for future use.

Users can define and edit topics of interests (i.e. thematic categories). For each topic of interest, a set of keywords that describe its content should be provided. Topics and keyword descriptions can be altered anytime. The retrieved Web pages can be saved for future reference in folders that correspond to the defined topics of interest. Those folders have a role similar to *Favorites* or *Bookmarks* in Web browsers.

Figure 4 shows the administration options for manag-

Figure 4. Administrating topics of interest.

ing topics of interest.

3. Automatic Classification of Retrieved Web pages

Captain Nemo recommends relevant topics of interest to classify the retrieved pages, exploiting nearest-neighbour classification techniques. The description of a retrieved Web page includes its title and a part of its content (which is usually its first few lines). The description of a topic of interest includes a set of keywords given by the user. The classification algorithm identifies the most relevant topic of interest for all retrieved pages, considering the description of retrieved Web pages and pre-defined topics of interest.

Classification Algorithm. *Captain Nemo* exploits Nearest Neighbor (Witten et al., 1999) as its main classification algorithm. The algorithm needs to calculate similarity measures between the description of each retrieved Web page and the description of every topic of interest. The similarity measure employed is a $tf-idf$ one (Witten et al., 1999). Let D be the description of a topic of interest and R the description of a retrieved Web page. The similarity between the topic of interest and the retrieved Web page, $Sim(R, D)$, is defined as follows:

$$Sim(R,D) = \frac{\sum_{t \epsilon R \cap D} w_{R,t} \times w_{D,t}}{\sqrt{\sum_{t \epsilon R \cap D} w_{R,t}^2} \times \sqrt{\sum_{t \epsilon R \cap D} w_{D,t}^2}} \quad (1)$$

where t is a term, $w_{R,t}$ and $w_{D,t}$ are the weights of term t in R and D respectively. These weights are:

$$w_{R,t} = \log\left(1 + \frac{C}{C_t}\right) \quad (2)$$

$$w_{D,t} = 1 + \log f_{D,t} \quad (3)$$

where C is the total number of topics of interest, C_t is the number of topics of interest including term t in their description and $f_{D,t}$ is the frequency of occurence of t in description D.

Having a new, retrieved Web page, we rank the topics of interest according to their similarity with the page (the topic of interest with the highest similarity will be on the top). Then the top-ranked topic of interest is selected as the most appropriate for the retrieved page.

Example. Let us assume that a user has the following three topics of interest: (t1) Sports: sports football basketball baseball swimming tennis soccer game, (t2) Science: science scientific mathematics physics computer technology and (t3) Arts: arts art painting sculpture poetry music decorating.

The result "Alen Computer Co. can teach you the art of programming...Technology is just a game now...computer science for beginners" receives the following similarity scores for each topic of interest:

$$Sim(x, t_1) = 0.287$$

$$Sim(x, t_2) = 0.892$$

$$Sim(x, t_3) = 0.368$$

The highest score corresponds to $t2$. Consequently, the most relevant topic of interest is "Science".

4. System Implementation

This section presents the architecture of our application and discusses various interesting implementation issues. Figure 5 describes the main modules of *Captain Nemo*.

Search Module. It implements the main functionality of the metasearch engine, providing connections to the search engines specified by the users. It retrieves the relevant Web pages according to the retrieval model defined by the user. The results are sent to the ranking module for further processing.

Ranking Module. The retrieved Web pages are ranked and grouped according to the retrieval model defined by the user. The ranking algorithm is presented in the next section. For every retrieved Web page, a matching topic of interest is determined.

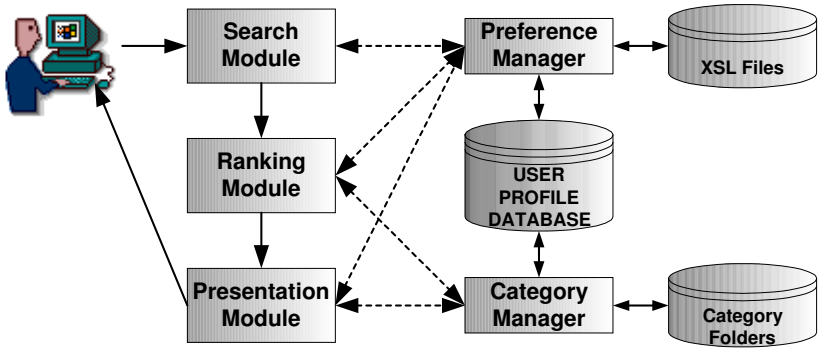

Figure 5. System architecture.

Presentation Module. It maintains several XSL filters that construct the resulting pages (wrapped as XML documents) according to the look and feel settings determined by the user.

Preference Manager. It provides the connection between the three aforementioned modules (i.e. search module, ranking module, presentation module) and the information stored in user profiles. It is also responsible for updating user profiles and the used XSL files.

Category Manager. It manages the topics of interests, keeps the appropriate folders on disk in accordance with the user profiles and provides all the necessary information for the automatic classification of results to those folders.

Our application is implemented on top of the PostgreSQL database system[4], exploiting Perl CGI scripts to wrap the results of search engines[5].

The next subsection discusses in detail the ranking mechanisms used in our application.

4.1. Ranking

Given a query, a typical metasearch engine sends it to several search engines, ranks the retrieved Web pages and merges them in a single list. After the merge, the most relevant retrieved pages should be on top. There are two approaches used to implement such a ranking task. The first one assumes that the initial scores assigned to the retrieved pages by each one of the search engines are known. The other one does not have any information about those scores.

[4]http://www.postgresql.org/

[5]http://search.cpan.org/dist/WWW-Search/lib/WWW/Search.pm

In Rasolofo et al. (2001), it is pointed out that the scale used in the similarity measure in several search engines may be different. Therefore, normalization is required in order to achieve a common measure of comparison. Moreover, the reliability of each search engine must be incorporated in the ranking algorithm through a weight factor. This factor is calculated separately during each search. Search engines that return more Web pages should receive higher weight. This is due of the perception that the number of relevant Web pages retrieved is proportional to the total number of Web pages retrieved as relevant for all search engines exploited by the metasearch engine.

On the other hand, Dumais (1994), Gravano and Papakonstantinou (1998) and Towell et al. (1995) stress that the scores of various search engines are not compatible and comparable even when normalized. For example, Towell et al. (1995) notes that the same document receives different scores in various search engines and Dumais (1994) concludes that the score depends on the document collection used by a search engine. In addition, Gravano and Papakonstantinou (1998) points out that the comparison is not feasible not even among engines using the same ranking algorithm and claims that search engines should provide statistical elements together with the results.

In Aslam and Montague (2001), ranking algorithms are proposed which completely ignore the scores assigned by the search engines to the retrieved Web pages: bayes-fuse uses probabilistic theory to calculate the probability of a result to be relevant to the query, while borda-fuse is based on democratic voting. The latter considers that each search engine gives votes in the results it returns, giving N votes in the first result, $N-1$ in the second, etc. The metasearch engine gathers the votes for the retrieved Web pages from all search engines and the ranking is determined

democratically by summing up the votes.

The algorithm adopted by *Captain Nemo* is the weighted alternative of Borda-fuse. In this algorithm, search engines are not treated equally, but their votes are considered with weights depending on the reliability of each search engine. These weights are set by the users in their profiles. Thus, the votes that the i result of the j search engine receives are:

$$V(r_{i,j}) = w_j * (max_k(r_k) - i + 1) \qquad (4)$$

where w_j is the weight of the j search engine and r_k is the number of results rendered by search engine k. Retrieved pages that appear in more than one search engines receive the sum of their votes.

4.2. Application Examples

The main page of *Captain Nemo* is illustrated in Figure 6. It includes the results of query 'perl', presenting only titles in compact format according to the user profile defined.

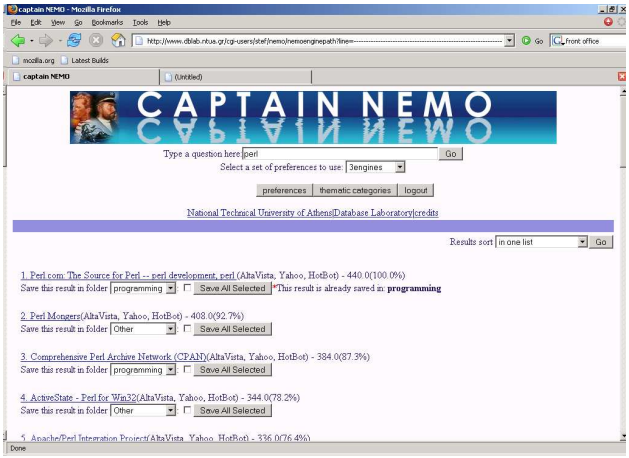

Figure 6. Captain Nemo.

Figure 7 shows the same results formatted by another presentation style. According to the preferences set, the results are merged in one list. For each retrieved Web page, we can see (a) the title, the description and the URL, (b) the names of search engines that have retrieved this particular page and (c) the absolute and relative similarity score calculated by the ranking module. A topic of interest is suggested for each retreived Web page.

Figure 8 shows the results for keywords 'java sql', grouped by topic of interest.

Figure 7. Retrieved Web pages for the keyword 'perl'.

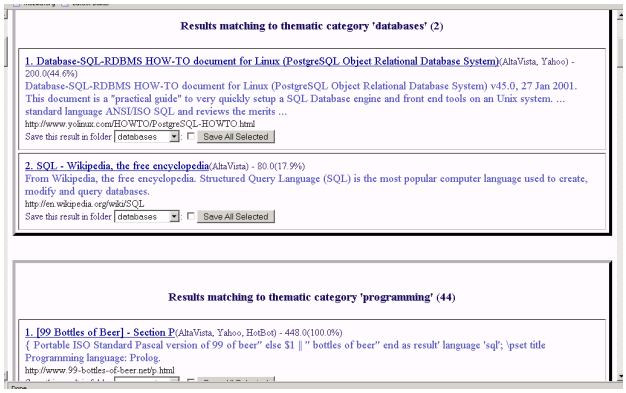

Figure 8. Retrieved Web pages grouped by topic of interest.

5. Conclusion

In this paper we presented *Captain Nemo*, a fully-functionable metasearch engine that exploits personal user search spaces. Users can define their personal retrieval model and presentation style. They can also define topics of interest. *Captain Nemo* exploits several popular Web search engines to retrieve Web pages relevant to keywords given by the users. The resulting pages are presented according to the defined presentation style and retrieval model. For every page, *Captain Nemo* can recommend a relevant topic of interest to classify the page, exploiting nearest-neighbour classification techniques.

For future work, we plan to replace the flat model of topics of interest by a hierarchy of topics in the spirit of Kunz and Botsch (2002). Also, we will improve the classification process, exploiting background knowl-

edge in the form of ontologies (Bloehdorn & Hotho, 2004).

References

Aslam, J. A., & Montague, M. (2001). Models for metasearch. *Proceedings of the 24th ACM SIGIR Conference.*

Baker, L. D., & McCallum, A. K. (1998). Distributional clustering of words for text classification. *Proceedings of the 21st ACM SIGIR Conference* (pp. 96–103). Melbourne, Australia: ACM Press.

Bloehdorn, S., & Hotho, A. (2004). Text classification by boosting weak learners based on terms and concepts. *Proceedings of the 4th ICDM Conference* (pp. 331–334).

Buntine, W. L., Löfström, J., Perkiö, J., Perttu, S., Poroshin, V., Silander, T., Tirri, H., Tuominen, A. J., & Tuulos, V. H. (2004). A scalable topic-based open source search engine. *Proceedings of the ACM WI Conference* (pp. 228–234).

Chakrabarti, S., Dom, B. E., & Indyk, P. (1998). Enhanced hypertext categorization using hyperlinks. *Proceedings of the ACM SIGMOD Conference* (pp. 307–318). Seattle, US: ACM Press, New York, US.

Cohn, D., & Hofmann, T. (2001). The missing link - a probabilistic model of document content and hypertext connectivity. *Proceedings of the 15th NIPS Conference.*

Dumais, S. T. (1994). Latent semantic indexing (lsi) and trec-2. *Proceedings of the 2nd TREC Conference.*

Glover, E., Flake, G., Lawrence, S., Birmingham, W. P., Kruger, A., Giles, C. L., & Pennock, D. (2001). Improving category specific web search by learning query modifications. *Proceedings of the SAINT Symposium* (pp. 23–31). San Diego, CA: IEEE Computer Society, Los Alamitos, CA.

Gravano, L., & Papakonstantinou, Y. (1998). Mediating and metasearching on the internet. *IEEE Data Engineering Bulletin, 21.*

iProspect (2004). iProspect search engine user attitudes. http://www.iprospect.com/premiumPDFs/iProspectSurveyComplete.pdf.

Kunz, C., & Botsch, V. (2002). Visual representation and contextualization of search results-list and matrix browser. *Proceedings of the ICDC Conference* (pp. 229–234).

Liu, F., Yu, C., & Meng, W. (2002). Personalized web search by mapping user queries to categories. *Proceedings of the 11th CIKM Conference* (pp. 558–565). McLean, Virginia, USA: ACM Press.

Rasolofo, Y., Abbaci, F., & Savoy, J. (2001). Approaches to collection selection and results merging for distributed information retrieval. *Proceedings of the 10th ACM CIMK Conference.*

Sahami, M., Mittal, V. O., Baluja, S., & Rowley, H. A. (2004). The happy searcher: Challenges in web information retrieval. *Proceedings of the 8th PRICAI Conference* (pp. 3–12).

Shahabi, C., & Chen, Y.-S. (2003). Web information personalization: Challenges and approaches. *Proceedings of the 3rd DNIS Workshop.*

Towell, G., Voorhees, E. M., Gupta, N. K., & Johnson-Laird, B. (1995). Learning collection fusion strategies for information retrieval. *Proceedings of the 12th ICML Conference.*

Witten, I. H., Moffat, A., & Bell, T. C. (1999). *Managing gigabytes: Compressing and indexing documents and images.* Morgan Kaufmann Publishers. 2nd edition.