Personalized Search with Editable Profiles

Binyam A. Zemede
Department of Computer Science
Texas State University
e-mail: bzemede@txstate.edu

Byron J. Gao
Department of Computer Science
Texas State University
e-mail: bgao@txstate.edu

***Abstract*—Search personalization is an important technique for improving search performance. Existing approaches work in a *black box*, where users have no clue how it works and how to customize it. This lack of user control and flexibility can often be inconvenient and counter-productive. In this paper, we propose PEEPLER, a *transparent* search personalization framework that enables full user control and manipulation. In PEEPLER, a user can own multiple profiles and each can be modified arbitrarily. Profile terms can be automatically generated, manually entered, and expanded by adding their semantically related ones. In addition, negative terms are allowed for specification of negative preferences, which can be very useful in filtering undesirable results. The selected profile will help re-rank search results based on how consistent they are with respect to the profile. We implement PEEPLER in the context of Web search using Google Web search API, demonstrating the promise and potential of the approach.**

Keywords-search personalization; user control; editable profiles; information retrieval; web search;

I. INTRODUCTION

With the unprecedented amount of information available from large corpuses and the Web, information retrieval systems and Web search engines are facing never-ceasing challenges in satisfying user needs. *Search personalization* allows tailoring and fine-tuning of search results based on individual preferences. It has become an important technique for improving search performance and drawn increasing attention from the information retrieval and data mining communities [2, 4, 7, 9, 10, 11, 13, 14, 15, 16]. Currently major Web search engines such as Google (googleblog. blogspot.com/2007/02/personally-speaking.html) and Yahoo (myweb.yahoo.com/) also provide such services.

Search personalization techniques typically build user profiles from search history indicating user preferences [5, 6, 12]. Existing approaches automatically extract and maintain such profiles and work in a *black box*, where users have no clue how personalization is done and how to customize it. This lack of user control and flexibility can often be inconvenient and even counter-productive. For example, such profiles usually capture long-term preferences and would fail to produce good results for short-term momentary information needs [1, 17]. In addition, multiple categories of preferences, if used together, would compete and work against one another for a particular query, given that queries are inherently and increasingly ambiguous. While it is impractical to build perfect user profiles, it is wise to introduce user intervention, open up the black box, and give users the privilege to directly and clearly specify their preferences. This motivates our PEEPLER project that investigates personalized search with editable profiles.

PEEPLER provides *transparent* search personalization, allowing direct and arbitrary user control. Note that it is not to replace, but to complement, existing automatic approaches and can work on top of them. In PEEPLER, a user can own multiple profiles. Each profile consists of a set of weighted keywords or terms. The terms can be added manually by users or automatically (not the focus for this implementation) from historical queries.

To completely cover the term space for a user interest and better match relevant search results, PEEPLER incorporates *semantic term expansion*, where the added terms are expanded by a set of semantically similar terms using WordNet [3]. In addition, users can assign negative weights to terms in order to specify negative preferences, which can be greatly useful in filtering undesirable search results. Users can pick any profile for personalized search. The selected profile has an influence to the ranking of search results, where search results that are more consistent with the profile will be considered more relevant and ranked higher, and vise versa.

Consistency is measured by using the traditional vector space model [8], where profiles and documents are represented as vectors of term weights, and cosine similarity is used to compute the similarity of each document with the selected profile. The similarity values indicate the degree of relevance of search results with respect to the profile. To demonstrate the promise and potential of our approach, we have implemented PEEPLER (www.peepler.info) in the context of Web search using Google Web search API (developers.google.com/web-search/docs). This is a work-in-progress project. The implementation is still under development and the current version well serves the purpose of proof of concept.

II. OVERVIEW

In this section we overview the PEEPLER framework and implementation.

Interface. Figure 1 shows a snapshot of PEEPLER. The interface provides a standard Web search layout with an additional profile selection down box. A logged-in user can easily specify any of her existing profiles for personalized search, or none of them for regular search. The "Manage profiles" tab allows the user to create, modify, and delete profiles. Pagination is enabled on the results with 10 results per page.

978-1-5386-2715-0/17/$31.00 ©2017 IEEE

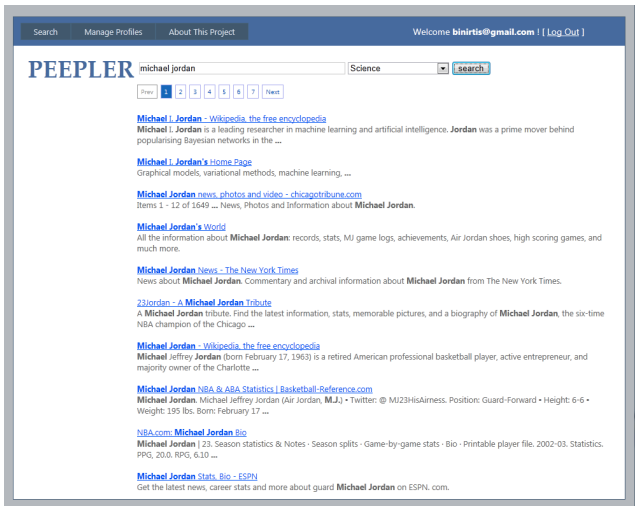

Figure 1. Snapshot.

Architecture. Figure 2 shows the main architecture of PEEPLER. It consists of three modules: profile management module, query processing module and personalization module. The query processing module takes a query q and retrieves a list R of search results from the search engine API. The profile management module allows the user to edit and maintain multiple profiles. The personalization module re-ranks the search results in R based on a chosen profile P. In the following, we explain the profile management and personalization modules in details.

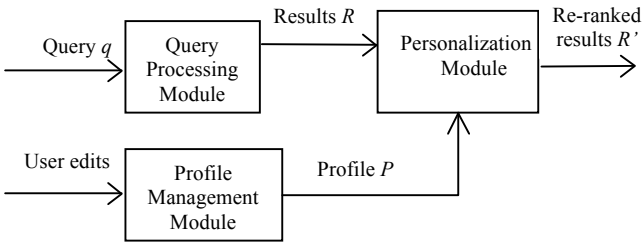

Figure 2. Architecture.

Profile management module. The profile management module allows a user to create, modify and delete multiple profiles. Semantically, a profile represents a specific category of user interests, e.g., "Science" or "Sports". Syntactically, it consists of a set of weighted terms, where weights range from -10 to 10. Positive weights indicate likes and negative weights indicate dislikes. Larger magnitudes indicate stronger interests.

The term space is large even for a specific area of interest. It is impossible for a user to enter sufficient amount of interesting terms, so that relevant search results can be well captured. PEEPLER solves this problem by *semantic term expansion*, where semantically similar terms to the entered terms will be automatically added to the profiles.

Semantic term expansion is done off-line using WordNet (wordnet.princeton.edu) [3]. Given a term t and a distance threshold d_0, PEEPLER automatically retrieves a set of terms that are within the semantic distance d_0 from t. When creating a profile, the user will get to choose d_0. In our experiments, when $d_0 > 2$, the search performance actually deteriorates. Thus currently PEEPLER only takes d_0 values that are smaller or equal to 2.

The weight of a newly added term t' depends on its distance d from the original term t in the semantic similarity tree. Let w be the weight for term t, the new term t' will be assigned a weight of w', where w' is computed using the following simple formula.

$$w' = w / d$$

To give users complete editing privilege, the added terms can also be managed at the term level. For example, these terms can be deleted and their weights can be adjusted.

To demonstrate that PEEPLER can incorporate and work on top of existing black-box personalization approaches, we also include a relatively simple automatic term generation component in PEEPLER. Briefly, with user permission, PEEPLER tracks user search activities, in particular, search terms, to generate profile terms. Search frequency is used to weight such terms. As in the case of semantic term expansion, users still have full editing privilege and control of these automatically collected profile terms.

Personalization Module. The personalization module adopts the conventional vector space model [8] to re-rank the search results in R returned by the query processing module. For efficient processing, the documents in R consist of snippets and titles only, instead of full documents. Previous research on clustering engines [18] has shown that snippets are as good as full documents. Although the applications (clustering vs. re-ranking) differ, both rely on similarity computation utilizing the vector space model. The documents in R are parsed and linguistically preprocessed by applying stemming, lemmatization, and stop word removal. Then, the TF-IDF (term frequency-inverse document frequency) weighting scheme is used to weight the terms, and each document is represented as a vector of TF-IDF term weights. TF-IDF is a standard term weighting scheme. Interested readers please refer to [8] for more details.

Similarly, profiles are considered as documents and represented as vectors of term weights. The difference is that the profile term weights are user-specified instead of computed using TF-IDF. In addition, the profile vectors are processed off-line instead of on-line as in the case of search results.

The PEEPLER personalization module then applies cosine similarity to compute the similarity for each search result and the selected profile. Larger similarity values indicate greater relevance of search results with respect to the profile. The original list of search results are re-ranked based on the similarity values. Specifically, we identify each pair of search results (r, r') such that the similarity value of r with respect to the profile is significantly larger than that of r'. Then we consider (r, r') as a confirmed preference (r is preferred to r') with confidence and it will be enforced in the re-ranked list. In other words, if r' is ranked higher than r in the original list, they will be swapped. The reason we use this re-ranking approach instead of simply ordering search

results by similarity values is that the original list embeds static ranking factors such as authority of Web pages, and we want to respect such factors while correcting the obvious relevance disorder.

III. Implementation and Usage

PEEPLER is implemented as AJAX enabled ASP.NET 4.0 Web application. It uses SQL Server as its backend database. Server side code is written in C#.NET. Client side code is written in Javascript with jQuery Javascript Library. The Web application is hosted on an IIS 7 Web server running Microsoft Windows Server 2008. PEEPLER provides Web search service using Google AJAX search API (developers.google.com/web-search/docs). The API can retrieve a maximum of 8 results per request and a total of 64 results per query.

You may use PEEPLER as a regular search engine without logging-in. However, in order to take advantage of search personalization, you will have to login in.

After logging in, you may create a profile (e.g. "Science") in the profile management section. You may specify a distance threshold (e.g., 1) and allow PEEPLER to perform semantic term expansion. Then, you may start entering preferred terms with desired weights (the default value is 5) into the profile. For example, you may enter a term "computer" with a weight of 8 and "biology" with a weight of 5. PEEPLER will perform semantic term expansion simultaneously for the terms you entered. For example, "calculator", "machine", "data", etc. will be added for the term "computer".

Now let's say you issue a query of "Michael Jordan", for which you intend to learn about the famous computer scientist *Michael Jordan*. Without using any profile, the results returned by the Google search API ranks Michael Jordan's (the famous basketball player) Wikipedia page first and his NBA profile page second. Actually, the top 10 results are all about the basketball player. The first result about your desired *Michael Jordan* is located on page 6 ranked as the 61st result. Now, issue the same query again after selecting the "Science" profile. You will see that PEEPLER re-ranks the Google result and *Michael Jordan*'s home page is ranked as the first result.

To see how negative terms help improve precision, now let's consider a "Shopping" profile that you have just created. Let's assume you have bad experience with Bestbuy (We love Bestbuy!) and have added a term "bestbuy" with weight -10 to the profile. If you issue a query "laptop" without choosing any profile, Bestbuy's laptop homepage will be ranked as the first result. But if you issue the same query after selecting the "Shopping" profile, then the Bestbuy laptop homepage will be displayed at the very end of all search results.

IV. Conclusion

We have proposed PEEPLER, a novel framework for transparent search personalization enabling full user control of profiles. Preliminary experiments have shown significant performance gain. Although we implement PEEPLER in the context of Web search, the principle can be applied to information retrieval systems in general. For future work, we plan to conduct comprehensive experiments and user studies to validate the framework. This implementation of PEEPLER is meant to demonstrate the promise of the general framework. We realize that there are many directions we can continue to improve PEEPLER. For example, how to organize profile terms for easier management, how to improve semantic term expansion for more complete coverage of user interests, and how to seamlessly incorporate existing automatic profile construction techniques for improved utility.

References

[1] P. Bennett, R. White, W. Chu, S. Dumais, P. Bailey, F. Borisyuk and X. Cui. Modeling the Impact of Short- and Long-Term Behavior on Search Personalization. In SIGIR, 2012.

[2] Z. Dou, R. Song, and J. Wen. A large-scale evaluation and analysis of personalized search strategies. In WWW, 2007.

[3] Christiane Fellbaum (1998, ed.) WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press.

[4] G. Jeh, and J. Widom, Scaling Personalized Web Search. In WWW, 2003.

[5] D. Jiang, K. Leung, and W. Ng, Context-Aware Search Personalization with Concept Preference. In CIKM, 2011.

[6] F. Liu, C. Yu, and W. Meng, Personalized Web Search by Mapping User Queries to Categories. In CIKM, 2002.

[7] Z. Ma, G. Pant, and O. Sheng. Interest-based personalized search. ACM Transactions on Information Systems (TOIS), 25 (1): Article 5, 2007.

[8] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008.

[9] A. Micarelli, F. Gasparetti, F. Sciarrone, and S. Gauch. Personalized search on the World Wide Web. The Adaptive Web, 4321: 195-230, 2007.

[10] A. Pretschner and S. Gauch. Ontology based personalized search. In proceedings of the 11th International Conference on Tools with Artificial Intelligence (TAI), 1999.

[11] F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In WWW, 2006.

[12] F. Radlinski and S. Dumais, Improving Personalized Web Search using Result Diversification. In SIGIR, 2006.

[13] X. Shen, B. Tan, and C. Zhai. Implicit user modeling for personalized search. In CIKM, 2005.

[14] A. Sieg, B. Mobasher, and R. Burke. Web search personalization with ontological user profiles. In CIKM, 2007.

[15] M. Speretta and S. Gauch. Personalized search baed on user search histories. In proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI), 2005.

[16] J. Sun, H. Zeng, H. Liu, Y. Lu and Z. Chen, CubeSVD: A Novel Approach to Personalized Web Search. In WWW, 2005.

[17] O. Zamir and O.Etzioni. Web document clusgtering: A feasibility demonstration. In SIGIR 1998.