Collaborative Re-Ranking of Search Results

Boris Chidlovskii, Natalie S. Glance and M. Antonietta Grasso

Xerox Research Centre Europe
6 chemin de Maupertuis
38240 Meylan, France
{chidlovskii | glance | grasso}@xrce.xerox.com

Abstract

In this paper we present a system architecture for coupling user and community profiling to the information search process. The search process and the ranking of relevant documents are accomplished within the context of a particular user or community point of view. The user and community profiles are built by analyzing document collections put together by the users and by the communities to which the users belong. Such profiles are used for ranking the documents retrieved during the search process. In addition, if the search results are (implicitly or explicitly) considered as relevant to the user or community, the user/community profiles can be tuned by re-weighting the profile terms.

Both recommender systems and meta-search engines can be enhanced using this kind of collaborative environment for ranking the search results and re-weighting the user and community profiles.

Introduction

In information retrieval, one principal problem is how to rank the results returned by a search engine or a combination of search engines. For individual search engines, there are many techniques for ranking results, ranging from counting the frequency of appearance of terms in the search query to calculating vector similarities between the term vector and the document vectors.

In a networked environment like the World-Wide Web, meta-search engines access different and often heterogeneous search engines and face the additional difficulty of combining the ranking information returned by engines. As many search engines hide the mechanism used for the document ranking, the problem of merging the results becomes even more difficult [1]. In addition, these kinds of approaches suffer from ignoring or knowing nothing about the user conducting the search, nor the context of the search.

Relevance feedback is one approach that elicits information about the user and his/her search context [2]. Based on user feedback, relevance feedback techniques re-rank the search results by re-calculating the relative importance of keywords in the query. While powerful from a technical point of view, relevance feedback approaches suffer from user interface issues: the relevance information required is difficult to successfully elicit from users during the search process.

In this paper, we propose to couple user and community profiling with the search process. Instead of being an isolated event, the search process is accomplished within the context of a particular user, community or point of view. The user and community profiles are built by analyzing document collections put together by the users and the communities to which the users belong. These document collections can take any (or a combination) of several forms: e.g., recommender system document collections, document management system document collections, personal bookmarks and/or file system document collections.

In the next section, we describe an architecture and system in support of community-based relevance feedback, the interfaces to already existing components that are required, and the details of the novel components, namely the search pre- and post-processor, and the user and community profilers. This system is currently under implementation.

System and Architecture

The basic elements of an architecture in support of community-based relevance feedback are:

- one or many document collections with associated user, community/group, and possibly rating attributes;
- one or many search engines or meta-search engines;
- a user profiler;
- a community profiler;
- a community manager;
- a search pre-processor for establishing the context of the search; and
- a search post-processor for ranking the combined set of returned results.

In addition, there may be wrappers that allow the profilers to extract document content (or document reference, such as URL), user, community and rating information from the document collections as well as wrappers that allow the search pre-processor to submit queries to (meta-)search engines and the search post-processor to extract the results.

AAAI-2000 Workshop on AI for Web Search

Document collection

The document collection(s) can be of several types: documents residing in document management systems, file systems, or referenced by recommender systems. In all of these cases, the document collections provide an implicit way of associating content with users and potentially with communities of users. The most useful document collections in the context of this approach are those provided by community recommender systems [3,4] which attach user ids, community categorizations, and user ratings to documents. This allows the most sophisticated forms of the user and community profiling techniques described below. Ideally, a community-based relevance feedback system based on the architecture described would include a recommender system as one of its document collections, if not as the principal one. However, it is still possible to build the system without one, although the notion of community is admittedly likely to become fairly weak.

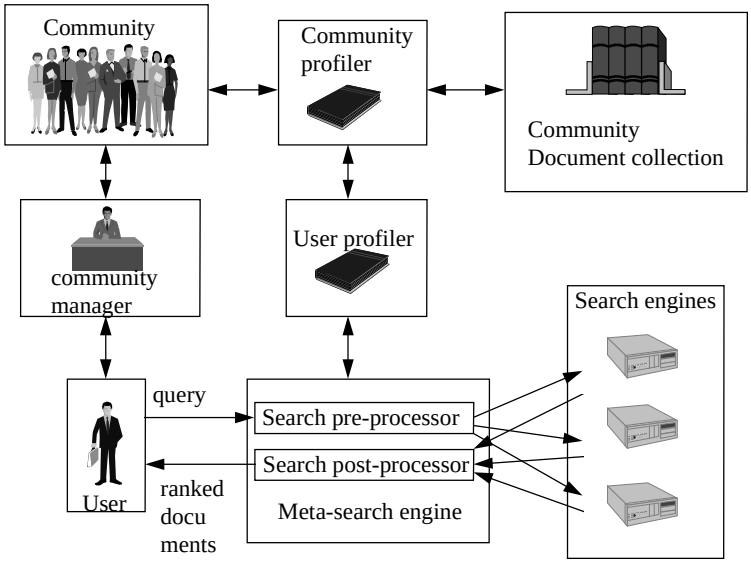

Figure 1 An architecture for community-based relevance feedback.

The document collection may possibly provide an API allowing the profilers to query for all documents submitted and/or reviewed by a user (possibly associated with a community). If such an API is not provided, then a wrapper is required that is able to extract the information.

Using this kind of meta-data, the profilers can construct and incrementally update user and community profiles from the set of documents relevant to a user and in the context of the community. In the case of standard document collections (file systems, document management systems), we assume that any document filed by a user is relevant. In the case of a recommender system, we assume that any document submitted or reviewed by a user with an average or higher rating is relevant.

(Relevance and quality are differentiated in a community recommender system. Relevance is a measure of pertinence for the community; quality is a measure of value to the community. Within the context of collaborative re-ranking, we convolve these two properties of documents. Well-rated documents do not receive a good rating generally unless they are at least relevant – thus we assume they are relevant and the higher the rating, the stronger the relevance. Poorly-rated documents, on the other hand, may be either judged as irrelevant to the community or of poor quality and as such are poor candidates for accurately providing negative relevance feedback.)

Search engine(s)

The search engine may or may not provide an API for submitting a search and retrieving results. If there is no API, a wrapper will need to built. The problem of query translation across multiple, heterogenous search engines and databases and the extraction of the search results is well-understood [5].

Note that the search engines do *not* necessarily cover the documents in the system's document collections! (Although serendipitous overlap is always possible.)

Search pre-processor

The search pre-processor is required to obtain the context of the user's search: the user's identity, the community or set of communities appropriate to the search, and possibly the point of view the user wishes to adopt for the search (another user -- for example, a domain expert).

The context can be retrieved either explicitly, by asking the user to identify him/herself and select the appropriate community or communities and/or point of view. Alternatively, it could be deduced automatically by matching the query with the query memory associated with a community (if selected) or the collection of people using the system.

User profiler

The user profiler constructs a term-weight vector for each user, extracted from the set of documents submitted and/or reviewed into each of the document collections to which the user participates. One difficulty is matching a user across several document collections. Currently, the only feasible way to do this is to ask the user to provide his/her identifier (and potentially password!) for each of these document collections. If the user withholds some of this information, then his/her profile will be less complete (but not necessarily worse – perhaps some document collections will be judged by the user as more suitable for providing an accurate profile). This problem disappears if we consider only one document collection in the system – for example, the one provided by the community recommender system.

The term-weight vector is calculated in a standard way, although various linguistic-based enhancements are possible and suggested below. For the user *u*, the vector contains the set of terms $\{t_i\}$ with their weights w_i^u. If the term-weight vector is at least in part calculated from documents that have been evaluated (implicitly or explicitly rated) in some way by the user, then the ratings given to the documents can be used to bias the term-weight vector.

The user profiler could also calculate the profile of the user in the context of a community or a specific domain or domains. In this case, the user profiler would take into account only those documents submitted or reviewed by a user and classified (either by the user or automatically) into the domain. An added difficulty in this case is matching communities/domains across document collections. Again, if the document collection is a recommender system, the difficulty disappears.

The user profiler provides an API that returns a term-weight vector in response to a user identification and (optionally) a community/domain identifier.

Community profiler

The community profiler constructs a term-weight vector for each community, extracted from the set of documents classified into a community within each of the document collections.

The term-weight vector for the community is determined in a way analogous to that employed for users. The community vector contains the set of terms $\{t_i\}$ with weights w_i^C. The weight of each term is calculated from the weights w_i^u of the community members (users). As contributions of the members are often much different, the community profile can be biased to weigh more heavily the contribution of "experts" in the community. Experts are those community members whose recommendations are mostly frequently followed by the whole community. Formally, each member *u* in the community is assigned with weight α_u; experts have the highest weights α_u and $\sum_u \alpha_u = 1$ over the community. (These weights must be re-normalized whenever someone enters or leaves the community.) Then, the weight of term t_i in the community profile is evaluated as $w_i^C = \sum_u \alpha_u w_i^u$, where w_i^u is the weight of t_i in the profile of user *u*. Beyond the community and personal profile, the user can request for the profile of the community expert(s), which contains weight $w_i^{\exp}$ for each profile term t_i.

The community profiler provides an API method that returns a term-weight vector in response to a community identifier.

Community manager

One principal difficulty is matching community definitions across document collections and maintaining a coherent list of communities and users participating in those communities. If a community recommender system exists as a component of the collaborative re-ranking system, then its list of communities will most likely be adopted for the community relevance feedback system as a whole. Potentially, the administrator of the system could be responsible for matching groups or collections in other document collections with the community list. The task of constructing such a list from scratch would fall to the administrator in the absence of such a list. It is also possible to devise automatic ways to do the matching, although this would reduce the accuracy of the community profiles.

It is possible that the way in which the community list is constructed and matched across collections will result in a community profile that is entirely determined from the data in the community recommender system, if such exists, or from another document collection with a notion of community.

Note that the system can still function in the absence of community profiles. It is still interesting to take into account the user's context, even in the absence of community, although it will be harder to deal with new users.

Search post-processor

The search post-processor takes as input the list of search results returned by the search engines. It has two ways to evaluate the relative rank of documents. One is by matching the document or its pointer (e.g., URL) to one already existing in one of the document collections. For example, if the document has been rated by a community recommender system connected to the system within the appropriate community context, then this rating is given a high weight in determining the relative rank of the result.

The second way to evaluate the relative rank of documents is by using the profile term-weight vectors as a source of relevance feedback. Depending on the context (user, community or expert), the appropriate profile is requested. Then, for each document, the content of the documents is retrieved, a term frequency vector is extracted, and the standard relevance feedback technique is employed to calculate the rank of the document. For each document *d* in the response to query *q*, the document rating is evaluated as follows (the adapted *cosine* function):

$$relevance_{q,d} = \frac{\sum_t w_{d,t} \times w_t^{prof} \times w_{q,t}}{W_d},$$

where $w_{d,t}$ is the weight of term *t* in the response document *d*, w_t^{prof} is the weight of term *t* in the chosen profile (user's w_t^u, community's w_t^C or expert's $w_t^{\exp}$), $w_{q,t}$ is the weight of the term *t* in the query *q*, and W_d is the vector length "projected" on the chosen profile and evaluated as $W_d = \sqrt{\sum_t (w_t^{prof} w_{d,t})^2}$. Note that weights $w_{d,t}$ are evaluated from the responses only. As responses contain often a brief description of original documents, the weight $w_{d,t}$ may be quite different from the term weights used by search engines storing the full documents. Also, the relevance rating is biased by the profile, through the terms w_t^{prof}.

Re-weighting user profiles

If the user gives a positive feedback to the search result or documents retrieved over the search process, the search results can be included into the document collection like any other recommendation. Moreover, the search result can be used to modify the user profile by re-weighting term weights. In such a case, the query terms and/or most frequent terms in the response form the set *Rel* of *relevant* terms. Using this set, we adopt the standard Rocchio formula for the relevance feedback. The main difference between our approach and standard relevance feedback is that we do not take into account *non-relevant* terms as we do not have a reliable way to extract this kind of information from both document collections and search results. As result of re-weighting, the relevant terms from *Rel* have their weights increased in the user profile.

Finally, the change of user profiles, accumulated over time, will trigger the update of the community profile and the weights of community experts.

Discussion

Our proposed system for collaborative re-ranking of search results provides two major advances on top of current meta-search engines:

- the ability to rank results returned across search engines;
- the ability to take into account the user's search context through use of user, community or expert user profiles.

However, there are drawbacks to the approach we propose. Namely, the time needed for result re-ranking may be critical parameter. The search post-processor requires textual content in order to evaluate the comparative relevance of the returned items in the context of a given user or community profile. This means downloading either an abstract (if available) or the entire document. The entire process could become very lengthy especially if the number of documents returned by the query is large. A first step of pre-filtering will be necessary in order to prune the list to a manageable number. However, while the time cost is high, it should be remembered that the time cost to the user of evaluating the returned documents him/herself is even higher, so in many cases users may be willing to turn collaborative ranking on, return to other work at hand, and await an alert indicating the collaborative ranking process has terminated. As a further user incentive to use the collaborative ranking feature, the items downloaded in the process can be cached locally, so that subsequent browsing by the user will be much less time-consuming. Alternatively, "short" profiles can be used for evaluating immediate yet approximate ratings.

Another issue is that document content comes in many formats. In order to operate across as many formats as possible, the search post-processor will need to be able to connect to other modules that transform different content formats into ASCII (when possible).

Finally, some documents will fall outside the system's ability to rank: the UI will need to distinguish these from those that are ranked in a way salient to the user.

Related work

In the situation of explosive growth of the Web data, gathering information on the Web becomes a more difficult problem. The personalization of the users' Web practice is one of the key approaches to facilitate this task. It includes two groups of methods, activity-based and content-based ones.

The activity-based methods personalize a user's web experience by joining personal activities with global information to effectively tailor what the user does. Web Browser Intelligence [6] personalizes the user practice done by organizing agents on a user's workstation to observe user actions and proactively offer assistance; it can annotate and record pages viewed for later access.

Similarly, the information about user preferences can be collected through cookies files in the Web [7]; these preferences are reused later to rank information delivered to the user.

Alternatively, search for Web documents can be personalized also by observing the iterative process of query preparation [8]. When a person has to formulate a query in the context of such a large document collections as the Web, this usually is an iterative process. By using certain induction methods, [8] measures how the starting query and the final result are related and, if it is justified, can link them each to other.

The alternative to activity-based approaches are content-based and collaborative methods used for personalized information filtering and document recommendations. These approaches are in part validated by the study reported in [9] which found that using profiles automatically created from documents ranked by the user as relevant actually outperformed profiles hand-crafted by users. Recommender systems, such as Fab and GroupLens, act between users and information resources; they maintain profiles of all users and operate in large document spaces.

GroupLens [10] applies the collaborative filtering to NetNews, to help people find articles they will like in the news stream. News reader clients display predicted scores and make it easy for users to rate articles; ratings are gathered and disseminated to predict scores based on the heuristic that people who agreed in the past will probably agree again. Fab [11] combines both content-based and collaborative filtering systems. It maintains user profiles based on content analysis and compares these profiles to determine similar users for collaborative recommendation. This approach is closest to what we propose in this paper, with the principal difference being that our field of application is search as opposed to WWW page discovery.

References

1. Gravano, L., Chang, C.-C.~K., Garcia-Molina, H. and Paepcke, A.. STARTS : Stanford Proposal for Internet Meta-Searching. In *Proceedings ACM SIGMOD'97*, pp. 207-218, 1997.
2. Salton, G. and Buckley, Ch. Improving Retrieval Performance by Relevance Feedback. In *Readings in Information Retrieval*. Eds. K.S.Jones, P. Willett, San Francisco: Morgan Kaufmann Publishers, 355-364, 1997.
3. Glance, N., Arregui, D. and Dardenne, M. Knowledge Pump: Supporting the Flow and Use of Knowledge. In *Information Technology for Knowledge Management.* Eds. U. Borghoff and R. Pareschi, New York: Springer-Verlag, 35-45, 1998.
4. Glance, N., Arregui, D. and Dardenne, M. Making Recommender Systems Work for Organizations. In *Proceedings of PAAM'99* , London, UK, April 1999.
5. Chidlovskii, B., Borghoff, U. and Chevalier, P.-Y. Boolean Query Translation for Brokerage on the Web, In *Proceedings EUROMEDIA'98 Conference*, Leicester, UK, January 1998.
6. Barrett, R., Maglio, P. & Kellem, D. How to Personalize the Web, In *Proceedings ACM CHI'97*, 75-82, 1997.
7. Meng, X. and Chen, Z. Improve Web search accuracy using personalized profiles. URL:http://www.cs.panam.edu/~meng/unixhome/Research/DataMine/Writing/spects99.ps, 1999.
8. Chalmers M., Rodden, K. and Brodbeck, D. The order of things: activity-centred information access, *Computer Networks and ISDN Systems*, 30, 359-367, 1998.
9. Foltz, P.W. and Dumais, S.T. Personalized Information Delivery: An Analysis of Information Filtering Methods. *Communications of the ACM*, 35(12): 51-60, 1992.
10. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. GroupLens: An Open Architecture for Collaborative Filtering of Netnews, Internal Research Report, MIT Center for Coordination Science, March 1994. URL:http://www-sloan.mit.edu/ccs/1994wp.html
11. Balabanovic, M. and Shoham, Y. Fab: Combining Content-Based and Collaborative Recommendation. *Communications of the ACM*, 40(3): 66-72, 1997.