Using Web Helper Agent Profiles in Query Generation

Gabriel L. Somlo and Adele E. Howe
Computer Science Department
Colorado State University
Fort Collins, CO 80523, U.S.A.
{somlo, howe}@cs.colostate.edu

ABSTRACT

Personalized information agents can help overcome some of the limitations of communal Web information sources such as portals and search engines. Two important components of these agents are: user profiles and information filtering or gathering services. Ideally, these components can be separated so that a single user profile can be leveraged for a variety of information services. Toward that end, we are building an information agent called *SurfAgent*; in previous studies, we have developed and tested methods for automatically learning a user profile [20]. In this paper, we evaluate alternative methods for recommending new documents to a user by generating queries from the user profile and submitting them to a popular search engine. Our study focuses on three questions: How do different algorithms for query generation perform relative to each other? Is positive relevance feedback adequate to support the task? Can a user profile be learned independent of the service? We found that three algorithms appear to excel and that using only positive feedback does degrade the results somewhat. We conclude with the results of a pilot user study for assessing interaction of the profile and the query generation mechanisms.

Categories and Subject Descriptors

H.3.3 [**Information storage and retrieval**]: Information Search and Retrieval—*Query formulation, Search process, Information filtering*

Keywords

information agents, user modeling, query generation

1. INTRODUCTION

The Web has become an indispensable source of information for many people. Based on surveys of the most popular Web sites [14], users deal with the overwhelming amount and constantly updating nature of the information by routinely visiting hub sites (e.g., Netscape, Yahoo, CNN) and making copious use of search engines (e.g., AltaVista, Excite, Magellan). Users have derived tremendous leverage from shared information resources such as those just mentioned. Hubs or portals provide communally useful information about perennial (e.g., financial management, child rearing) and timely (e.g., September 11 events, stock quote) topics. Search engines satisfy specific, spontaneous information needs.

As our appetite for information increases, so does its availability on the Web. Studies (e.g., [21, 12]) have identified limitations with these tools for satisfying users' needs; for example, users appear to lack the motivation to learn how to formulate complex queries or to view long lists of potential matches. Meta-search engines, such as Meta-Crawler [18], SavvySearch [6], and NECI [11], propose to overcome the Web coverage problem by combining the indexing power of multiple stand-alone search engines. However, because they leverage the capabilities of many search engines, they tend to generalize the search task: limiting the access to search-engine-specific advanced search capabilities and, perhaps, introducing even more noise into the return results.

One promising approach to compensating for the limitations is to *personalize* the tools. Pretschner and Gauch divide personalization into two types: personalized access to resources and filtering/ranking [15]. For example, *My Yahoo* (http://my.yahoo.com) provides personalized access by allowing users to design their own Yahoo page with pertinent information; many search and meta-search engines support some customization (e.g., types of search, return amount, and search engine selection). "Softbot"s or information agents augment searching and information gathering (filtering/ranking). Personalized information agents, such as Letizia [13], WebWatcher [1, 10], and WebMate [5], can provide a range of services from automatically retrieving Web pages to assisting in formulating queries.

These agents generally comply with the architecture presented in Figure 1. The agent intercedes between the user and their Web access, monitoring the user's activities to construct a model of user requests (the profile) to be used for specific information services (e.g., modifying requests and/or filtering results). In our research, we adopt the principle that user overhead needs to be minimized:

- The profile should be learned by asking the user to simply click on a feedback button positioned on the bottom of each page to indicate interest.
- Learning should track changes in user interests.
- The profile should support multiple information services.

AAMAS'03, July 14-18, 2003, Melbourne, Australia

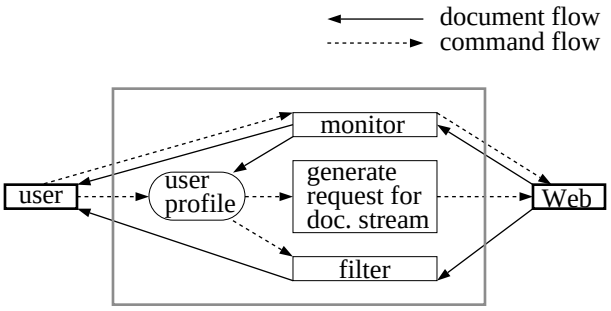

Figure 1: Architecture of a Web information helper agent

In previous papers, we have assessed some alternative approaches to learning user profiles [20, 19]. In this paper, we examine alternative approaches to one of the services: query generation to find new documents (i.e., automatically retrieving Web pages that match a user's interests by submitting queries to a Web search engine). In particular, we are interested in answering the following questions:

1. *How do different algorithms for query generation perform relative to each other?* For our case, query generation involves constructing queries from a user profile that are submitted to a search engine for the purpose of harvesting documents.

2. *Is positive relevance feedback adequate to support the task?* To minimize user overhead, we have solicited only *positive* relevance feedback. Obviously, this provides relatively weak knowledge, requiring the profiling mechanism to self-organize the categories of interest and to trade-off precision.

3. *Can a user profile be learned independent of the service?* If so, then user overhead can be minimized and multiple services provided as part of the same agent.

This paper describes a framework for evaluating alternative algorithms for information gathering agents and a study that was designed to address the three questions above. In summary, we found: Three algorithms perform best in terms of extracting sufficient numbers of documents from the search engine and in terms of the relevance of the links returned. We did find evidence that soliciting only positive feedback hampers query generation; however, it is not clear that the degradation in performance is worth the cost of obtaining the negative feedback. As often happens, the study raised some issues that are still to be resolved (particularly about the evaluation criteria and the interaction of profiling and query generation); we conclude with a pilot study in which we investigate how to resolve these issues.

2. SURFAGENT

SurfAgent [19] is a personalized Web information agent, which follows the basic architecture in Figure 1. It is designed as a testbed for expediting plug-and-play and evaluation of alternative algorithms, front-ends, and service tasks. Its two primary components are the user profile and the module which generates requests for document streams. Monitoring should be simple and unobtrusive. Filtering depends on the representation and construction of the user profile, forcing a relatively tight coupling of those two components. This section provides an overview of its user profiling and document stream generation.

2.1 Building User Profiles and Filtering

The user profile maintained by a Web helper agent is a model of what documents the user finds relevant. Like most other personal Web helper agents, SurfAgent uses TF-IDF vectors [17] as the basis of its user profile representation. One such vector is used to represent each of the several different topics of interest associated with each user.

Over time, topic descriptions are learned from user-supplied examples of relevant documents, which are added to the existing TF-IDF vectors in what is known as *relevance feedback* [16]. Associated with each vector is a *dissemination threshold*, which is used when new documents are filtered by the agent: if the similarity between the new document's vector and a vector in the profile exceeds the associated dissemination threshold, the document is considered relevant and shown to the user. We found that learning the appropriate dissemination threshold was critical to filtering performance and that one could be learned with relatively little feedback (i.e., 10 relevance judgments) [19].

TF-IDF vectors and their associated dissemination thresholds are known in the Information Retrieval (IR) literature as *filtering queries*. This type of query is distinguished from a typical retrieval query (used with search engines or at a library) by a few characteristics. Filtering queries tend to be used repeatedly over a long period of time, during which they can be improved and maintained through learning and relevance feedback, whereas retrieval queries are typically used only once. Also, filtering queries typically contain lots of terms with complex weighting schemes, whereas retrieval queries tend to be a boolean combination of relatively few terms, with no weighting at all.

Each filtering query in SurfAgent's user profile corresponds to a distinct topic of interest. User profiles are learned in one of two ways. First, relevant documents provided as training by the user can be explicitly associated with the topic of interest. Alternatively, to minimize overhead to the user, incremental clustering can be used by the agent to automatically classify relevant examples into internally learned topics [20, 5]. In the latter situation, the user only needs to prompt the agent when a relevant document has been encountered, without having to associate it with a particular topic of interest. To minimize user disruption, we request only positive examples. We augmented existing IR clustering techniques to accommodate Web needs (i.e., avoid storing the documents themselves, require minimal user overhead and be associated with a user). In our earlier study, we found that a tuned version of the Doubling algorithm [4] achieved high recall, without a great sacrifice on precision.

2.2 Incoming Document Streams

Personal information agents use a wide range of techniques to generate incoming streams. Letizia pre-fetches Web pages by exploring the links in the Web page currently being viewed. Similarly, WebWatcher analyzes text in and around links in the current page to predict relevant links worth following. Fab builds a list of likely to be relevant documents through best-first search; documents that pass the filtering phase are then included in a list of recom-

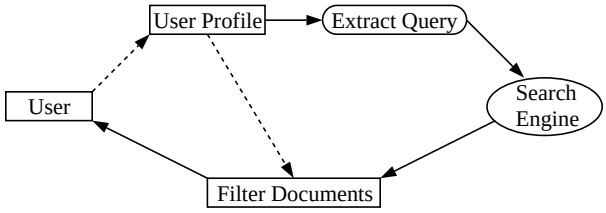

Figure 2: Incoming document streams generated by querying a search engine

mended links. Finally, WebMate filters articles found on a list of well-known news sources in order to compile a personal newspaper for its user.

Our goals are to maximize the quality of the incoming document stream generated for SurfAgent, while at the same time minimizing effort. For this purpose, a promising technique appears to be the construction of queries that are suitable for a large-scale search engine such as Google [3]. Well-formulated queries have the potential to off-load significant portions of the filtering task to the search engine, which should provide the agent with a document stream consisting of more relevant documents.

In this paper, we explore several methods of generating search engine queries from the user profile of a personal Web helper agent. We wish to find both the method and the query length that would optimize the relevance of the documents returned by the search engine with respect to the user profile. This process is illustrated in Figure 2.

3. TECHNIQUES FOR QUERY GENERATION

Filtering queries are not directly suitable for being submitted to a search engine. They are complex models representing a possibly large collection of documents; they contain a large number of terms with associated weights, which would overly restrict the range of documents a search engine might return.

Query generation techniques have evolved from the more general query refinement mechanism of relevance feedback [16]. For instance, in [9, 7] search engine queries are extended with features extracted from positive and negative examples in order to bias them toward a more relevant subtopic. Several other researchers have been concerned with extracting only a few highly representative terms from the representation of a large document cluster. For WebACE [2], the authors propose a mechanism for generating queries from a cluster of documents based on the average word count (term frequency, TF) of the terms contained in the documents of the cluster, and the document frequency (DF), i.e., the number of documents in the cluster that contain a specified term. A set of k terms with the highest TF, and another set of k terms with the highest DF are selected from the cluster. The intersection of these two sets was submitted to Yahoo search as a query, yielding a total of 2280 results, which were later refined to 372 by augmenting the query with terms resulting from the difference between the TF and DF sets.

CorpusBuilder [8] uses automatic query generation to collect documents in a specified language (Slovenian) from the Web, with the purpose of building a corpus in that language. The point is to preserve computation power by avoiding a brute-force crawl of the Web where an expensive classifier would have to be run on each encountered document. By generating queries from the already existing corpus, the authors hope to significantly increase the likelihood that the resulting documents would already be in Slovenian, thus speeding up document collection. Several methods for generating queries are used interchangeably:

- uniform – select n terms from the relevant documents with equal probability;
- term-frequency – select n most frequent terms from the relevant documents;
- probabilistic term-frequency – select n terms from the relevant documents with probability proportional to their term frequency;
- odds-ratio – select n terms with highest odds-ratio scores, as given by the formula:

$$OR_t = \log_2 \frac{P(t|relevant) \cdot (1 - P(t|nonrelevant))}{P(t|nonrelevant) \cdot (1 - P(t|relevant))}$$

  where t is the term, and $P(t|relevant)$ and $P(t|nonrelevant)$ are the probabilities of t appearing in a relevant and non-relevant document, respectively;
- probabilistic odds-ratio – select n terms with probability proportional to their odds-ratio score;

The authors report best results with $n = 4$ and the simple odds-ratio method. However, this method is not necessarily applicable to our task because identifying relevance with respect to a query cluster is somewhat more subtle than determining whether a returned document is in a particular language such as Slovenian.

4. OUR STUDY OF QUERY GENERATION

The purpose of this study is to examine the role of query generation technique for a Web information agent: what techniques work well, how much user overhead is warranted and how query generation interacts with profiling. These three factors correspond to the three questions articulated in the Introduction.

4.1 Experiment Design

The basic protocol for our study was as follows:

1. construct user profiles,
2. generate queries from those profiles using each of the query generation methods,
3. submit queries of different lengths to Google,
4. evaluate the results.

This led to a factorial experiment in which the independent variables were query generation method and query length and the dependent variables were two evaluation metrics: return count and relevance.

4.1.1 Constructing the profiles

To expedite experimental control and data collection, we constructed two user profiles from TREC[1] disk #5. TREC data consist of a large number of articles (over 250,000), of which a large portion have been judged as either relevant or non-relevant with respect to a set of 450 different topics. Our first topic is on airport security measures, and was constructed based on 46 relevant documents which appeared in the Los Angeles Times over the course of two years (1989-1990); this topic will be referred to as LATIMES. The second topic is on genetic research, and was constructed based on 55 relevant documents from the Foreign Broadcasting Information Service appeared in 1996; this topic will be referred to as FBIS. One topic was picked randomly from each of the two document collection on the TREC disk.

We used synthetically generated topics in order to test the hypothetical scenario in which negative feedback is available. By default, SurfAgent does not collect negative feedback in the form of documents which are *non*-relevant to a given topic. Thus, we are interested in how much performance might be sacrificed by restricting feedback to only positive examples. The number of positive documents used in the construction of each topic (46 and 55, respectively) is realistic compared to what a human user would have been capable of providing while building her profile in real life.

4.1.2 Generating Queries

We implemented several methods, including both methods which use such negative examples (e.g., odds-ratio) against methods which do not (e.g., Boley's method [2] and term frequency based methods).

In addition to the methods mentioned in Section 3, we add two methods: deterministic extraction of highest weight terms for SurfAgent's TF-IDF profile vectors and probabilistic weighted extraction from the TF-IDF profile vectors. The complete list of methods used is given below:

Uniform (Unif) baseline case, select n terms with uniform probability;

Boley select the intersection of the k top ranking terms according to term frequency in one set, and document frequency in the other;

TF select n top ranking terms according to term frequency;

Probabilistic TF (P-TF) select n terms with probability proportional to their term frequency;

OR select the top ranking n terms according to their odds-ratio score;

Probabilistic OR (P-OR) select n terms with probability proportional to their odds-ratio score;

TFIDF select n terms with the highest TF-IDF weight from the profile vector;

Probabilistic TF-IDF (P-TFIDF) select n terms with probability proportional to their TF-IDF weights;

[1]Text REtrieval Conference: TREC benchmark disks are publicly available and can be ordered from the conference homepage at `http://trec.nist.gov`

The probabilistic versions were included because injection of small amounts of randomness has been found to be helpful in other search problems.

Code from SurfAgent was modified to collect the data required by all query generation methods employed. For each topic, we collected the following data:

- average term frequencies for terms in relevant documents;
- document frequencies for terms in relevant documents;
- TF-IDF vector built from relevant documents;
- odds-ratio scores for terms in relevant documents (odds-ratio scores are based on both relevant and non-relevant documents related to the topic).

From these data, we generated queries of seven lengths (two to eight terms) for each of the eight methods. For the four probabilistic methods, we generated 10 queries of each length, which means their reported results will be averages over those 10 queries.

For Boley's method, we repeatedly computed the intersection of the sets consisting of the top k ranking terms w.r.t. TF and DF, while incrementing k. We retained all distinct queries of length between 2 and 8. For FBIS, no value of k generated a query of length 6. Similarly, for LATIMES, there was no value of k resulting in a query of length 7.

4.1.3 Submit the Queries

The previous step resulted in 614 queries (307 for each topic). We submitted these queries to the Google search engine and collected back the first page of results. By default, a page can contain up to 10 responses.

4.1.4 Collect the results

The results of the queries (the URLs returned) were parsed out of the page returned by Google, and their corresponding documents were retrieved from the Web and stored locally. We discarded (and decremented our hit count accordingly) all dead links and all hits that were in a format other than ASCII[2] or PDF: a total of 312 out of 2917 hits were discarded. The PDF documents were converted into ASCII using the `pdftotext` utility.

4.2 Results

For each valid hit, we computed the similarity between the document's TF-IDF vector and the TF-IDF vector of the appropriate topic, which is a measure of the document's relevance. For each combination of query generation method and query length, we recorded the number of hits received, the relevance of the best hit, and the average relevance over all hits received. For the probabilistic methods, these measurements represent average values obtained over the ten repetitions for each combination of parameters. The results are summarized in Table 1 for the FBIS topic, and Table 2 for the LATIMES topic. The three rows corresponding to each method indicate average relevance (top), maximum relevance, and number of returned hits (bottom).

All methods return at least seven documents with query lengths of 2, but most taper off in the number returned

[2]ASCII includes all variants and versions of HTML, XML, etc.

method		query length 2	3	4	5	6	7	8
Unif	avg:	.022	.046	.059	.018	—	—	.011
	max:	.051	.077	.101	.019	—	—	.011
	cnt:	7.7	4.3	3.2	0.4	0	0	0.1
P-TF	avg:	.026	.054	.044	.053	.069	.091	.082
	max:	.059	.096	.079	.102	.120	.192	.138
	cnt:	8.9	7.7	7.9	5.2	6.5	7.2	6.3
P-OR	avg:	.039	.047	—	.019	—	—	—
	max:	.099	.090	—	.019	—	—	—
	cnt:	9.0	3.2	0	0.1	0	0	0
P-TFIDF	avg:	.045	.058	.088	.069	.035	.034	.030
	max:	.100	.110	.178	.097	.054	.035	.055
	cnt:	9.1	6.1	8.4	2.4	2.7	0.6	1.4
Boley	avg:	.053	.077	.090	.081	—	.111	.088
	max:	.120	.112	.136	.168	—	.239	.239
	cnt:	9	9	10	7	0	8	9
TF	avg:	.036	.031	.048	.082	.081	.087	.083
	max:	.065	.059	.129	.134	.103	.130	.135
	cnt:	10	9	10	9	9	10	9
OR	avg:	.123	.186	.102	—	—	—	—
	max:	.155	.361	.190	—	—	—	—
	cnt:	9	8	2	0	0	0	0
TFIDF	avg:	.100	.144	.160	.176	.214	.278	.242
	max:	.377	.377	.377	.279	.399	.404	.242
	cnt:	9	10	10	7	10	4	1

Table 1: Average relevance, Maximum relevance, and count of returned hits for the FBIS topic on genetic technology

method		query length 2	3	4	5	6	7	8
Unif	avg:	.012	.012	.012	.013	.004	—	—
	max:	.024	.024	.028	.019	.006	—	—
	cnt:	8.0	5.3	3.9	1.0	0.6	0	0
P-TF	avg:	.017	.026	.025	.028	.032	.024	.010
	max:	.042	.073	.062	.061	.046	.042	.011
	cnt:	9.1	9.5	6.0	6.5	2.0	4.0	0.7
P-OR	avg:	.017	.018	.016	.011	—	.007	—
	max:	.052	.039	.029	.013	—	.007	—
	cnt:	8.2	8.3	4.0	0.9	0	0.1	0
P-TFIDF	avg:	.026	.036	.064	.063	.059	.020	.010
	max:	.058	.103	.125	.156	.106	.036	.014
	cnt:	9.2	8.1	8.1	5.7	5.3	1.3	0.2
Boley	avg:	.040	.098	.135	.193	.199	—	.167
	max:	.086	.199	.299	.343	.359	—	.299
	cnt:	8	9	8	8	8	0	7
TF	avg:	.107	.058	.030	.048	.041	.069	—
	max:	.222	.093	.051	.075	.069	.069	—
	cnt:	7	10	10	7	6	1	0
OR	avg:	.048	.036	.348	—	—	—	—
	max:	.122	.096	.402	—	—	—	—
	cnt:	9	9	2	0	0	0	0
TFIDF	avg:	.115	.144	.155	.171	.144	.153	.143
	max:	.331	.331	.357	.299	.276	.349	.349
	cnt:	9	7	8	8	9	9	9

Table 2: Average relevance, Maximum relevance, and count of returned hits for the LATIMES topic on airport security

with longer query lengths. For the deterministic methods, the relevance increases as the query length increases (until 7 or 8), but the relevance for the probabilistic methods tends to plateau early.

All methods consistently outperform the baseline uniform term selection. Probabilistic methods are outperformed by

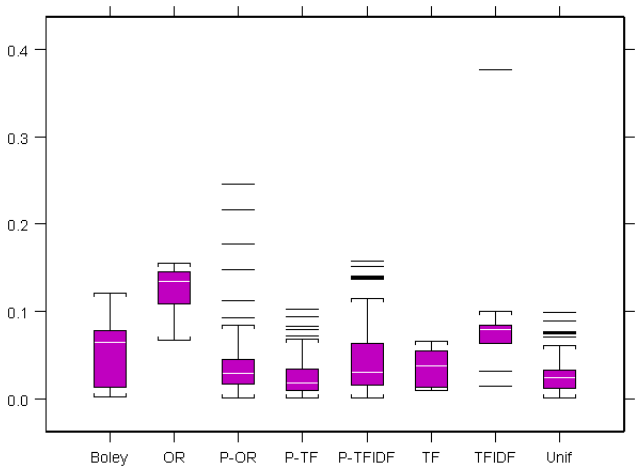

Figure 3: Box plot of relevance by method for FBIS topic at query length 2

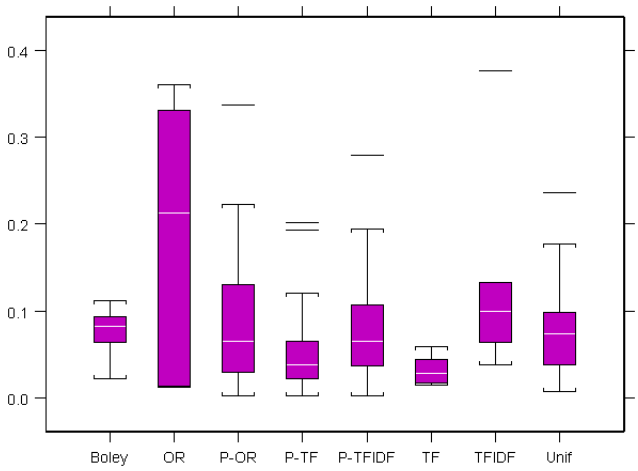

Figure 4: Box plot of relevance by method for FBIS topic at query length 3

the non-probabilistic ones, which is consistent with the observations in [8]. The best results for the FBIS topic were obtained using TFIDF at query length 7: a total of 4 hits were returned, with an average relevance of .278, and a maximum relevance of .404. The best results for the LATIMES topic were obtained using OR at query length 4: two hits were returned, with average relevance of .348 and maximum relevance of .402.

Query lengths 2 and 3 were the only ones where all methods lead to non-empty returns for both topics. To test whether the differences between the methods were significant, we ran an analysis of variance (ANOVA) study on each topic at query lengths 2 and 3, with the query generation method as the independent variable, and relevance as the dependent. The effects were significant in each case: for FBIS, we obtained $F = 14.007$, $p < 0.001$ at query length 2, and $F = 8.692$, $p < 0.001$ at query length 3; for LATIMES, we obtained $F = 24.027$, $p < 0.001$ at query length 2, and $F = 20.277$, $p < 0.001$ at query length 3.

Box plots of relevance by method for query lengths 2 and 3 are given in Figures 3 and 4 for FBIS, and Figures 5 and

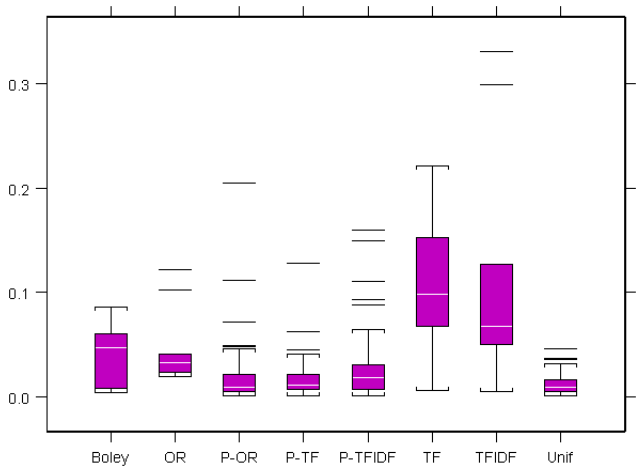

Figure 5: Box plot of relevance by method for LA-TIMES topic at query length 2

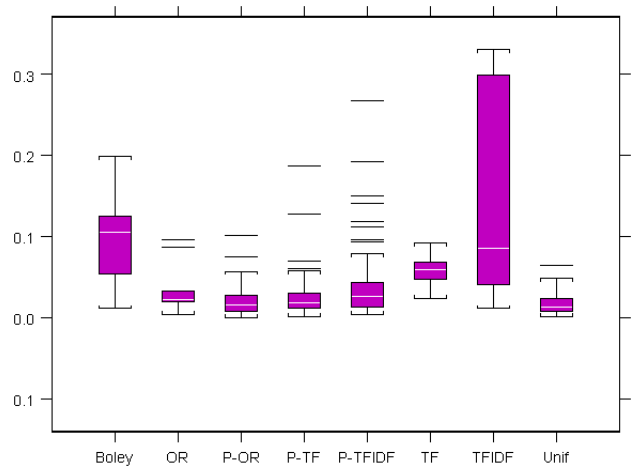

Figure 6: Box plot of relevance by method for LA-TIMES topic at query length 3

6 for LATIMES. Note that medians rather than means are marked on the graph. These plots illustrate several points obscured by the previous table. First, while TFIDF returns the best match and second best median in all cases, overall better results are obtained using OR for FBIS, and TF and Boley for LATIMES. Second, each method can show high variance in the results, although TFIDF tends generally to higher variance. Additionally, the results for query length 3 have higher variance than those for query length 2. Finally, the distributions are often skewed. For example, Boley consistently has a higher median than mean.

Because relevance for all returned documents is measured against the TF-IDF vector of the corresponding topic, the experiments are slightly biased in favor of the TFIDF query generation method. Our experiments cannot prove that TFIDF query generation is sufficient, but its good performance coupled with the not always good performance of OR suggest that we do not incur a significant loss by leaving out negative feedback. Collecting other information based on positive feedback in addition to TF-IDF topic vectors may be required with SurfAgent: e.g., straight TF vectors and

method		query length 2	3	4	5	6	7	8
TFIDF	nrel:	8	3	4	4	4	—	—
	cnt:	10	10	10	10	9	0	0
P-TFIDF	nrel:	1.0	1.8	0.7	0.4	0.0	0.5	—
	cnt:	9.5	8.1	7.7	4.1	1.3	1.3	0.0

Table 3: Number of relevant documents and count of returned hits for the user-generated topic on stress relief

topic-specific document frequency information would allow us to use TF and Boley query generation in addition to the TFIDF method. As the results show, sometimes TF and Boley perform better than OR and TFIDF.

Both Boley and TFIDF consistently result in many links being returned, even for long query lengths. Many hits are desirable when the agent will be pruning out previously visited sites and filtering the returned links according to a filtering threshold.

The computation time required for query generation by each of the studied methods is negligible when compared to the network-related delays incurred while downloading the actual documents.

5. PILOT USER STUDY

To gain an understanding of the performance of TFIDF query generation without the bias present in our experiments with synthetically generated topics, we have also performed a pilot study with a user-generated topic, containing 34 documents on stress relief. Since SurfAgent only collects TF-IDF information at this time, query generation was limited to the TFIDF and P-TFIDF methods. We followed the same protocol as with the synthetically generated topics: query lengths were between 2 and 8, and results were averaged over ten trials for the probabilistic version P-TFIDF. A total of 343 distinct URLs were returned by Google. We shuffled these URLs and asked the user to examine each one of them, and mark the ones she found to be relevant to her topic. 56 documents out of the total of 343 were found relevant. Table 3 presents the number of relevant documents and the number of hits returned for each parameter combination.

This pilot study supports the hypothesis that TFIDF based queries will generate an adequate incoming stream: queries of length up to six returned at least nine hits from Google. Unlike the previous study, the shorter queries yielded lower relevance, which could be due to the way the user was judging relevance or to the nature of the topic.

As a followup, we will be designing a user study that includes the three apparently best methods (TF, TFIDF, and Boley). We will focus on three issues: Does method performance vary among users and topics (as is suggested by our current study)? Should profile construction incorporate more information? Does relevance assessment change as profiles become more mature? Can the best query length be determined a priori?

6. CONCLUSIONS

We studied several methods for generating queries from a user profile to answer three questions related to the design of Web information agents. First, how do different query

generation algorithms perform relative to each other? In fact, we observed significantly different performance among the eight methods tested. Overall, Boley, TFIDF and to a lesser extent TF provided a good number of hits and relatively high relevance.

Second, is positive relevance feedback adequate to support the task? We found that leaving out negative training examples does not incur a significant performance loss. Odds-Ratio was found to excel on one topic, but its competitive advantage does not appear to be worth the additional overhead expected from the user. TFIDF and Boley, requiring only positive relevance feedback, generated queries that resulted in relevant hits.

Third, can user profiles be learned independently of the service? The results from TFIDF and the pilot experiment do suggest it. However, the pilot study also suggests that either user relevance judgments may be a bit lower (harsher) than the automated method or that the profile may not adequately reflect the users' interests. In fact, the good performance of Boley and TF indicates that in some cases it might be worthwhile to collect more than TF-IDF information from the user-supplied positive training examples. This last question will be examined in more detail in the future.

Our study confirmed that additional user burden in the form of negative feedback appears unwarranted to support document generation and that queries generated based on automatically learned profiles can guide harvesting of new documents of interest. This last result is excellent news for the development of agents that leverage a single learned profile to personalize a multitude of web information services.

7. ACKNOWLEDGMENTS

This research was supported in part by National Science Foundation Career Award IRI-9624058. The United States Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation herein.

8. REFERENCES

[1] R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell. WebWatcher: A Learning Apprentice for the World Wide Web. In *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Resources*, Stanford, CA, 1995.

[2] D. Boley, M. Gini, R. Gross, E. Han, K. Hastingsand G. Karypis, V. Kumar, M. Bamshad, and J. Moore. Document Categorization and Query Generation on the World Wide Web Using WebAce. *AI Review*, 13(5-6):365–391, 1999.

[3] S. Brin and L. Page. The Anatomy of a Large-scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, pages 107–117, 1998.

[4] M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental Clustering and Dynamic Information Retrieval. *Proceedings of the 29th ACM Symposium on Theory of Computing*, 1997.

[5] L. Chen and Katia Sycara. WebMate: A Personal Agent for Browsing and Searching. In *Proceedings of the Second International Conference on Autonomous Agents*, Minneapolis, MN, 1998.

[6] D. Dreilinger and A.E. Howe. Experiences with selecting search engines using meta-search. *ACM Transactions on Information Systems*, 15(3):195–222, 1997.

[7] G.W. Flake, E.J. Glover, S. Lawrence, and C.L. Giles Extracting Query Modifications from Nonlinear SVMs. In *Proceedings of the Eleventh International World Wide Web Conference (WWW 2002)*, Honolulu, HI, U.S.A., 2002.

[8] R. Ghani, R. Jones, and D. Mladenic. On-line learning for query generation: Finding documents matching a minority concept on the web. In *Proc. of the First Asia-Pacific Conference on Web Intelligence*, 2001.

[9] E.J. Glover, G.W. Flake, S. Lawrence, W.P. Birmingham, A. Kruger, C.L. Giles, and D. Pennock. Improving Category Specific Web Search by Learning Query Modifications. In *Proceedings of the IEEE Symposium on Applications and the Internet (SAINT 2001)*, San Diego, CA, U.S.A., 2001.

[10] T. Joachims, D. Freitag, and T. Mitchell. WebWatcher: A Tour Guide for the World Wide Web. In *Proc. of the 15th International Joint Conference on Artificial Intelligence*, Nagoya, Japan, 1997.

[11] S. Lawrence and C.L. Giles. Context and page analysis for improved web search. *IEEE Internet Computing*, 2(4):38–46, 1998.

[12] S. Lawrence and C.L. Giles. Searching the world wide web. *Science*, 280:98–100, April 3 1998.

[13] H. Lieberman. Letizia: An agent that assists web browsing. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, Montreal, Canada, 1995.

[14] Inc. Netmation. 100 most popular web sites. http://netmation.com/list100.htm.

[15] A. Pretschner and S. Gauch. Personalization on the web. Technical Report ITTC-FY2000-TR-13591-01, Dept. of Electrical Engineering and Computer Science, University of Kansas, December 1999.

[16] J.J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, 1971.

[17] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, 1988.

[18] E. Selberg and O. Etzioni. The metacrawler architecture for resource aggregation on the web. *IEEE Expert*, 12(1):8–14, 1997.

[19] G.L. Somlo and A.E. Howe. Adaptive lightweight text filtering. In *Proceedings of the 2001 Conference on Intelligent Data Analysis (IDA '01)*, Lisbon, Portugal, September 2001.

[20] Gabriel L. Somlo and Adele E. Howe. Incremental clustering for profile maintenance in information gathering web agents. In *Proceedings of the 2001 International Conference on Autonomous Agents (AGENTS'01)*, Montreal, Canada, May 2001.

[21] A. Spink, J. Bateman, and B.J. Jansen. Searching heterogeneous collections on the web: Behavior of excite users. *Information Research: An Electronic Journal*, 5(2), 1998. http://www.shef.ac.uk/~is/publications/infers