An Adaptive Web Page Recommendation Service

Marko Balabanović

Department of Computer Science,
Stanford University

Abstract

An adaptive recommendation service seeks to adapt to its users, providing increasingly personalized recommendations over time. In this paper we introduce the “Fab” adaptive web page recommendation service. There has been much research on analyzing document content in order to improve recommendations or search results. More recently researchers have begun to explore how the similarities between users can be exploited to the same ends. The Fab system strikes a balance between these two approaches, taking advantage of the shared interests among users without losing the benefits of the representations provided by content analysis. Running since March 1996, it has been populated with a collection of agents for the collection and selection of web pages, whose interaction fosters emergent collaborative properties. In this paper we explain the design of the system architecture and report the results of our first experiment, evaluating recommendations provided to a group of test users.

1 Introduction

When faced with an insurmountably large array of choices, people often turn to some kind of recommendation service. For instance they will look in their newspaper for movie reviews, or in a travel guide for tourist sights. We are interested in creating an automatic recommendation service which over time adapts to its users, so that they receive increasingly personalized recommendations. We can think of on-line recommendation as a three-stage process:

Collection First collect the items to be recommended.

Selection Next select from the collected items those best for a particular user.

Delivery Finally deliver the selected items to the user.

Most on-line recommendation or search services can be described in this way. For instance a web index (e.g., (Mauldin & Leavitt 1994)) will have an exhaustive collection component, a selection component which is an

information retrieval (IR) system to match web pages to user-supplied queries, and will deliver the pages via the web. A news clipping service collects items by subscribing to news wires, selects items for users according to profiles they submit, and might deliver them by fax or e-mail.

Our work explores adaptive methods for the collection and selection of web pages. In order to facilitate this exploration we have designed the “Fab” architecture as a test-bed for trying out different collection and selection methods, supporting a population of users and recording experimental results.

The two major classes of adaptive recommendation service are *content-based* (recommending items based on some analysis of their content) and *collaborative* (recommending items based on the recommendations of other users). We have attempted to establish a middle ground, taking advantage of the shared interests among users without losing the benefits of the representations provided by content analysis.

In this paper we report results from our ongoing first experiment. The Fab system has been instantiated with classes of adaptive content-based collection and selection agents, whose interaction fosters emergent collaborative properties. By “agent” we simply mean a process which maintains a long-term state—in our case a profile which can be used to induce a ranking among web pages. The operation of the system is as follows: users can request recommendations at any time, and will be shown the ten highest-ranking web pages according to their profile. They rate each page according to how well it matches their interests, and the collection and selection agents use this feedback to refine their profiles.

The following section places the work in context by describing related projects from both the IR and the AI communities. We then describe the design of the architecture and of the agents which populate it. Finally we describe our experimental design and give results from our first experiment.

2 Related Work

Two strands of related research are of particular relevance: work on document classification (which considers alternative techniques for selection) and collaborative systems (which propose a different architecture based on a novel selection technique).

2.1 Document Classification

Document classification lies at the intersection of machine learning (ML) and IR, and there is a large literature. To enable the application of ML techniques there has been much interest recently in dimensionality reduction (e.g., (Deerwester *et al.* 1990)). In the IR community variants of relevance feedback have been studied in the context of the routing task at the TREC conferences (Harman 1995), for example (Buckley *et al.* 1995; Allan 1995). There have also been numerous comparisons between these variants and non-incremental ML techniques (Schütze, Hull, & Pedersen 1995; Lang 1995; Pazzani, Muramatsu, & Billsus 1996). One of the disadvantages of such techniques is the large number of examples that are required before the learning algorithm can be applied. In contrast we assume an on-line model where the user gradually sees better and better pages, and we do not assume a fixed document collection—users’ feedback influences which pages are collected and shown to them.

Some of the work on document filtering (where the classification is used to pick relevant documents from an incoming stream) shares our on-line model of learning (Sheth & Maes 1993; Foltz & Dumais 1992). Assisted browsing systems (Armstrong *et al.* 1995; Lieberman 1995) share our goal of recommending web pages, but restrict themselves to the section of the web just ahead of the user’s current browser location, and then recommend appropriate links to follow.

2.2 Collaborative Systems

Collaborative recommendation services (Shardanand & Maes 1995; Resnick *et al.* 1994) present an interesting alternative to traditional IR techniques. Typically they work by identifying the nearest neighbors to a user in the space of past ratings, and then recommending something one of these neighbors has rated highly.

Domains with a constant stream of new items are difficult for collaborative systems: a single item may not be rated very often by different users, and much of the information gained by the system is rooted in the past and therefore unusable. Examples of such domains are web pages or news feeds. In contrast more static domains (e.g., music) or those with a much smaller number of new items appearing (e.g., movies) will enable collaborative systems to work well, as lots of different users will rate the same items. Additionally for these domains it is very hard to do any content analysis so content-based systems will be harder to implement.

3 System Design

3.1 Background

For the content-based approach which we are taking, there are four essential requirements:

w A representation of a web page.

m A representation of the user’s interests.

$r(\mathbf{w}, \mathbf{m})$ A function to determine the pertinence of a web page given a user’s interests.

$u(\mathbf{w}, \mathbf{m}, s)$ A function returning an updated user profile **m** given the user’s feedback s on a page **w**.

The assumption underlying content-based systems is that the content (rather than appearance, interactivity, speed of loading, etc.) of a page is what determines the user’s interest. Now we make the further assumption that we can represent the content of a page purely by considering the words contained in the text. We ignore all mark-up tags, images and other multimedia information.

The vector-space model of information retrieval (Salton & McGill 1983) provides us with an appropriate representation for documents based on their constituent words. This model has been used and studied extensively, forms the basis for commercial web search systems and has been shown to be competitive with alternative IR methods (Harman 1995).

In this model documents and queries are represented as vectors. Assume some dictionary vector **d**, where each element d_i is a word. Each document then has a vector **w**, where element w_i is the weight of word d_i for that document. If the document does not contain d_i then $w_i = 0$.

In our formulation we reduce words to their stems using the Porter algorithm (Porter 1980), ignore words from a standard stop list of 571 words, and calculate a TFIDF weight: the weight w_i of a word d_i in a document W is given by:

$$w_i = \left(0.5 + 0.5 \frac{tf(i)}{tf_{max}} \right) \left(\log \frac{n}{df(i)} \right)$$

where $tf(i)$ is the number of times word d_i appears in document W (the *term frequency*), $df(i)$ is the number of documents in the collection which contain d_i (the *document frequency*), n is the number of documents in the collection and tf_{max} is the maximum term frequency over all words in W .

The document frequencies are approximated by using a fixed dictionary of approximately 70,000 stemmed words created from a sample of 5,229 randomly picked web pages (so in the above equation $n = 5,229$). If d_i does not occur in the dictionary we set $df(i) = 1$.

In an attempt to avoid over-fitting, and reduce memory and communications load, we use just the 100 highest-weighted words from any document. Recent experiments described in (Pazzani, Muramatsu, & Billsus 1996) have shown that using too many words leads

Excellent
 Very Good
 Good
 Neutral
 Poor
 Very Bad
 Terrible

Figure 1: Ordinal scale users use to rank documents.

to a decrease in performance when classifying web pages using supervised learning methods. Our exploratory experiments have agreed with their findings that the optimum is between 30 and 100. We are using the figure of 100 pending more extensive experiments which will be possible once more user ratings have been gathered.

Once the top 100 words have been picked we normalize $\mathbf{w}$ to be of unit length, to allow comparisons between documents of different lengths.

This vector representation is used both for pages, as explained, and for the model of the user’s interests, the *user profile* $\mathbf{m}$ (which corresponds to the query in a retrospective IR system). In order to measure how well a page $\mathbf{w}$ matches a profile $\mathbf{m}$, we use a variant of the standard IR cosine measure:

$$r(\mathbf{w}, \mathbf{m}) = \mathbf{w} \cdot \mathbf{m}$$

Updating $\mathbf{m}$ also corresponds to a normal operation in retrospective IR: relevance feedback (Rocchio 1971). We use a simple update rule:

$$u(\mathbf{w}, \mathbf{m}, s) = \mathbf{m} + s\mathbf{w}$$

where s is the user’s score for page $\mathbf{w}$ (we translate the scale shown in Figure 1 to the integers 3, 2, 1, 0, -1, -2, -3). Relevance feedback has been demonstrated to be a powerful technique—experiments have shown that queries automatically generated through relevance feedback can outperform human searchers (Foltz & Dumais 1992; Harman 1995).

We assume that a user’s interests change over time, and furthermore that this happens in real time, regardless of the number of recommendations requested per day. This process is modelled using a simple decay function—each night all the weights in the profiles are multiplied by 0.97.

3.2 Architecture

Figure 2 shows the main components of the architecture: selection agents, collection agents and the central repository. All agents maintain a profile: each user has a selection agent, which maintains their user profile; each collection agent maintains a search profile which is used to guide it in its collection of web pages.

The goals underlying the design of the architecture were twofold:

- To allow scaling up, so that it would be easy to add extra users, agents and resources.
- To take advantage of potential overlaps between users’ interests.

The architecture is best explained by considering the path of a document W through the system. Let’s say the document is found by agent A , one of the various types of collection agent (more details in section 3.3), and immediately converted to its vector representation $\mathbf{w}$. At regular intervals collection agents submit the pages they have gathered which best match their search profiles to the central repository, replacing the pages they previously submitted. Thus at any time the repository contains each collection agent’s best pages in their own opinions.

When a user requests their Fab recommendations their selection agent (of which there is one per user) picks, from the entire repository, those pages which best match the user’s personal profile. The user then ranks these pages. These rankings are used both for evaluation purposes (discussed in section 4) and as feedback for the agents to learn from. The selection agent uses the feedback to update the user’s personal profile (using the function u). It also forwards the feedback, via the central repository, to the originating agent A , which will update its search profile in the same way.

There are several advantageous features of this architecture:

- A user’s personal profile is a valuable resource, representing considerable effort and time spent ranking pages. Its representation as part of the user’s selection agent means that the user’s information will never be lost through some adaptation scheme or “drowned out” by other users (it is only updated with the user’s own feedback). Furthermore it makes the profile available to the user for use with other applications (e.g., e-mail or news filtering), as opposed to in a purely collaborative system, where there is no meaningful way to consider one user’s profile in isolation.
- A brand new user to the system is shown a selection of pages which are randomly chosen from the repository. However the repository contains pages which various agents believe will best match the current user population. Thus the new user is already starting from a much higher level than would be expected from an empty profile, especially if the system is deployed in an organization or special interest group where there will be significant overlap between users’ interests.
- It is possible to cheaply maintain a population of “parasitic” users. We can add every evaluation received (from all users) into an *amalgamated profile*, which represents an average of the user community. We can then pick pages from the repository according to this profile and make them available, thus

Figure 2: Overview of the Fab architecture.

allowing “parasites” to view pages preselected by an established group of users, without any requirement to provide feedback¹.

In addition to these features, our design relies on the hypothesis that the following two behaviors will be exhibited:

Specialization By only giving feedback to the collection agent which originally found a page, we are attempting to encourage specialization. If an agent begins to specialize to a topic, users not interested in that topic should no longer be shown pages from that agent, which should narrow its specialization further. Eventually we expect the agent will settle into a niche where there is a group of users interested in the topic which its profile represents. In future experiments we hope to see the overlap between users’ interests leading to an economy of scale, whereby several users interested in one topic can be served by a single agent, and conversely users interested in several topics can be served by several agents.

Serendipity If any collection agent finds a page which matches a user’s profile, the user should see it. This includes agents who have specialized to finding pages for other groups of users, agents who monitor various existing web page recommendation services, agents which merely produce random pages, etc. This property can be thought of as allowing for serendipitous “word-of-mouth” recommendation, whereby the user is shown pages from outside of their regular sources which happen to match their interests.

So far we have only discussed the short-term learning of the agents: updating profiles based on users’

¹Top pages from Fab selected according to its amalgamated profile are available daily at <http://fab.stanford.edu>

rankings of pages. However there also exists a need for a longer term evolution of the agents, to insure a spread of profiles in the space of topics. It is undesirable to maintain collection agents which have very similar profiles (it would be more efficient to combine them), or agents whose pages are rarely picked or receive very low scores. We have defined our architecture so that this long-term adaptation component is easily replaceable, as we intend to conduct further research to investigate different schemes.

3.3 Collection Agents

We have implemented two kinds of adaptive collection agents, along with a variety of non-adaptive kinds for comparative purposes.

Search Agents First described in (Balabanović & Shoham 1995), these agents perform a best-first search of the web. For each agent the heuristic function used to evaluate a web page $\mathbf{w}$ is $r(\mathbf{w}, \mathbf{m})$, where $\mathbf{m}$ is the agent’s search profile. In order to better cope with the increasing size and branching factor of the web, we limit the size of the search fringe, and restrict the number of nodes from any particular site. A record of all nodes visited is maintained separately, with a one month timeout. In effect we have a hybrid best-first/beam search.

Index Agents Rather than actually searching the web, these agents attempt to construct queries for existing web indexes in an attempt to avoid duplicating work. The indexes used are Alta Vista², Inktomi and Excite. Since little information about the IR models employed by these commercial systems is forthcoming, it is hard to design optimal queries. Furthermore the

²Links to all the on-line services mentioned can be found on the web at <http://fab.stanford.edu/papers/aa97/>.

indexes are tailored mainly to short queries, and entering more than 20 search terms often results in zero recall. Thus the current implementation is fairly rudimentary: submit as a disjunctive query the top p words from the agent profile, where p has been optimized by hand for different indexes, and varies from 10 to 20.

Non-Adaptive Agents For purposes of comparison we have implemented some simpler agents which do not maintain their own profile:

NO-MEMORY INDEX AGENTS These agents work exactly like regular index agents, but they draw their words from the amalgamated profile. Thus these agents are not trying to specialize but are serving the “mass market”.

RANDOM AGENTS These agents retrieve pages from various sources of random pages available on the web (i.e. they are not truly randomly picked pages, but rather randomly picked from various preselected collections). Currently pages are drawn from Alta Vista, Yahoo, URouLette.

COOL AGENTS These agents retrieve human-recommended pages from nine “cool page of the day” sites around the web. These pages have been selected for their interest to the general community, not any particular user, and are often newly released sites.

Long-term evolution As explained in section 3.2, some mechanism to insure that agents’ profiles spread out in the vector space is required. This remains a topic for further research, but for the time being we have a preliminary implementation. The following procedure is followed separately for each agent type, once a week:

1. Find best and worst agents over last week, according to their median feedback score.
2. If they both scored worse than **Neutral**, restart the worst agent from scratch.
3. If the worst agent scored worse than **Neutral** but the best agent scored better, duplicate the best agent and kill the worst.

3.4 Selection Agents

Currently only one selection method is used, based on the comparison function $r(\mathbf{w}, \mathbf{m})$. We calculate the value of this function (using the user’s profile) for every page in the repository. The highest-scoring pages are shown to the user, with the proviso that no two are identical or from the same site, and that the user has not seen an identical page in the last month.

4 Experimental Design

4.1 Background

We have decided to use a new performance measure developed by Yao (1995) in our experiments, in prefer-

ence to the standard IR measures of precision and recall or the standard machine learning measure of classification accuracy. The *ndpm*³ measure is a distance between the user’s ranking of a set of documents and the system’s ranking of the same documents, normalized to lie between 0 and 1. Space precludes a definition, but we will summarize the advantages of this new measure.

The notion of relevance as used in IR is problematic in general, and in particular for our domain. User studies have shown that users have difficulty making consistent relevance judgments over a long period of time when asked to rate documents on an absolute relevance scale (Lesk & Salton 1971; Saracevic 1975), and in fact this is a general feature of human judgments (Rorvig 1988). There will also be considerable disagreement between the judgments of different users (Saracevic 1995). The usual way to circumvent this problem is to test IR systems on standardized collections of documents and queries, so as to make recall and precision figures comparable.

Since our domain is the web, we cannot rely on using a standardized collection. Furthermore, since there is no query involved in the use of our system the concept of relevance is inappropriate. Saracevic (1975) has drawn a distinction between the user’s unarticulated *information need* and the *query* which results from that. Relevance is relative to the query, whereas *pertinence* is relative to the information need. In our system it is unnecessary to formulate a query, so we can only attempt to measure how well the information need is met.

In contrast to the negative results from user studies mentioned, it has been shown that human judges are good at making *relative* judgments given a collection of documents, and will be consistent over time and compared to other judges (Lesk & Salton 1971).

The *ndpm* measure only requires relative judgments from users, and since there is no query involved the judgments relate to pertinence not relevance. Thus it is more appropriate than precision and recall, which are based on absolute relevance judgments. Furthermore, directly comparing the user and system rankings gives us a single-valued measure, simpler to interpret than precision-recall graphs (interestingly in the degenerate case where two-level rankings are used, *ndpm* can be shown to be a composite measure of recall and precision, and is related to several other measures proposed over the years—details in (Yao 1995)). Even apart from these considerations, recall is impossible to measure and difficult even to estimate when the document collection is the whole web.

An alternative approach would be to use classification accuracy as our performance measure. However as with precision and recall we would have to ask users to classify items on an absolute scale, which is prob-

³Normalized Distance-based Performance Measure

lematic as already discussed.

4.2 Selection Performance

On a day-to-day basis the system supplies the user with a number of documents it thinks the user will rank highly. It uses the resulting scores in order to perform relevance feedback and improve the user profile. Thus the documents shown to the user represent a narrow segment of the entire set of documents, close to each other in terms of user preference, making them an unsuitable set for use with the measure described. Furthermore during the normal operation of the system it has been observed that users will skew their rankings in order to influence the system's future behavior (i.e. the feedback they give does not represent their actual opinions).

Therefore we have used the following scheme. At regular intervals the user is given a special list of documents, and asked to rank them according to their own interests. This list is selected randomly from the repository, which contains documents produced by agents specializing in many different topics as well as agents who produce random pages. The rankings are not used for relevance feedback, so there is no ulterior motive for the user. The system also ranks the documents according to how well they match the current user profile (this is the system's prediction of the user's rankings).

We use an ordinal scale to obtain user rankings. The user places each document into one of seven categories as shown in Figure 1 (Cox (1980) recommends 5, 7 or 9-point scales; the adjectives are selected from (Mittelstaedt 1971)).

The desired result for this experiment is for the *ndpm* distance between the user's and system's rankings (averaged over all users at the same evaluation point) to decrease gradually over time, as the system's model of the user becomes more accurate.

4.3 Collection Performance

When discussing the use of his distance measure, Yao assumes that the entire document collection is ranked by both the user and the system. This is clearly impractical but our alternative formulation leaves us with a problem. It is possible for the system to learn a perfect model of the user without it ever recommending any pages the user wants to see (e.g., it could always correctly predict that the user's rating would be **Terrible**). Thus we need some way to measure the absolute performance of the system as well, to see how well the collection component is doing.

In order to do this we measure how well the system-recommended *personal* pages perform in comparison to pages from three other sources: *random* pages, *cool* pages and *public* pages. Random and cool pages are found by random and cool agents respectively; public pages are those pages from the repository which best match the amalgamated profile.

Computer graphics/Game programming
Library cataloging and classification
Post-industrial music
Sports information and gaming
Native American culture
Cookery
60's music
Hiking
Evolution

Figure 3: Topics chosen by users.

There has been criticism of collaborative recommendation services (Mitchell 1995) that they have never been shown to do better than the "majority vote" rule, i.e. they would do as well just recommending the most popular items. This is not directly applicable in our case: as there is a constant stream of new items, no one item will necessarily get rated by many users. However the comparison between personal and public pages gives us a measure of how much difference the personalization makes.

The methodology of the earlier experiment is easily adapted for this new comparison: we show users an equal number of pages from each source, randomly permuted. For each of the sources there is an ideal ranking consisting of two equivalence classes, whereby all of its pages are ranked above all other pages. Thus we can measure for each source the distance between this ideal ranking and the actual ranking supplied by the user (using the *ndpm* measure as before).

4.4 Declared Topics

Our most scarce resource is not computer speed or storage, but the attention and quantity of users willing to test the system. Thus we decided to try to combine the two experiments described into one, and furthermore to try to build up a dataset which could be used for future off-line experiments.

The results which follow describe a combined experiment. Each user was asked to declare a topic of interest in advance (Figure 3), and rate pages according to that topic. This was to allow easier interpretation of the profiles learned and help us re-use the data for future experiments. Every five rounds of evaluation they were shown a selection of 16 pages, four each from the four sources as described. The relative rankings of the sources **and** the difference between user and system rankings were recorded. We can think of these two measurements as recording the performance of collection and selection agents, respectively.

5 Results

For this series of experiments we ran nine best-first search collection agents for eight hours per day each, during which time they would inspect between 1,000

and 5,000 web pages (depending on computer and network load). We also ran two of each type of index agent (once every 8 hours), two random agents (also once every 8 hours) and one cool agent (once a day). As well as the users participating in the experiment we supported around 30 parasitic users, whose feedback did not affect the collection agents. The results are shown from 25 evaluations, corresponding roughly to one month of usage. Nine users participated in the experiment, although only seven made it to the 15th evaluation, only five to the 20th and only two to the 25th. All the profiles were started empty.

Figure 4: Average *ndpm* distance between user and system rankings, over all users at evaluation points.

5.1 Selection Performance

Figure 4 shows the average *ndpm* at each evaluation point. The gradual downward progression indicates that the selection agents are successfully learning increasingly accurate profiles of the users over time, as the distance between the predicted and actual rankings decreases. Indeed by the 25th evaluation it is down to almost nothing.

The difficulty of the topics chosen by users varies enormously. A very rough idea of this can be gained by doing a simple keyword search of the Infoseek web index: 20,558 documents match “cooking” but only 158 match “library cataloging classification”. One of the advantages of the vector space representation is that we can inspect the learned profiles: 90% of the top 400 terms in the profile learned for cooking are obviously cooking related, compared to only 2% for library cataloging and classification.

5.2 Collection Performance

Figure 5 shows how pages from the different sources are performing relative to one another. As expected the personal pages improve as adaptation occurs, and do

significantly better than pages from the other sources. Given the small number of predeclared topics it is not surprising that the cool pages are not much better than the random pages. The public pages will provide a more interesting comparison when we allow additional users and arbitrary topics, as there will be a greater overlap of interests.

Figure 5: For each source, average *ndpm* distance between user rankings and its ideal ranking, over all users at evaluation points.

There was considerable variation in collection agents performance for individual users. For instance the user interested in cooking received between six and ten pertinent documents per day after only four days, whereas the user interested in library cataloging, despite receiving many documents about libraries, never received one on the exact topic.

A comparison of the *success* (median feedback score) and *popularity* (average percentage of users’ pages supplied) scores for search and index agents yields no discernible trends as yet, with both classes of collection agent performing similarly. The index agents use negligible resources in comparison to the search agents, but we expect over a longer period that the finer adaptability of the search agents will be a dominant factor.

5.3 Specialization

In some cases it is very easy to see the specialization that has gone on among the collection agents—for instance an inspection of all the profiles reveals that one agent is a “cooking expert”, with 77% of the top 400 terms obviously cooking related, one other has a tangential interest and the rest have no apparent interest. Indeed the cooking user customarily receives between 50% and 90% of their recommendations from the “cooking expert” agent.

On the other hand “music” is a common theme for two of the topics, and this is reflected in the

less strict specialization of the agents. Three agents have an approximately equal number of music-related terms in their profiles, and the two users interested in music-related topics typically receive their music-related pages from a mix of these agents.

5.4 Serendipity

An example of the serendipitous recommendations resulting from the interplay between selection and collection agents comes from a collection agent specializing in pages about India (mistakenly, as the intended topic was Native American cultures). Apart from attempting to serve the user interested in Native American cultures, at various times this agent provided pages on biodiversity in India to a user interested in evolution and recipes for Indian food to the user interested in cooking.

A problem we encountered during this experiment was users being extremely strict, giving a high score only to pages exactly on their topic and a very low score otherwise. This leads to a lack of positive examples to learn from and subsequent fairly random behavior. The system works best when users “guide” it, by starting with a broad topic and slowly narrowing it down. We are investigating ways to inform users about effective ranking strategies.

6 Conclusions

Fab has been running “live” since March 1996, and provides a robust test-bed for comparing adaptation techniques and investigating agent interactions. Our design shows how in a specialized domain it is possible for multiple agents to coordinate their activities without explicit communication. As in a collaborative system, a user’s recommendations are influenced by similar users of the system. However we retain the advantages of content-based systems: a user’s profile exists as a stand-alone representation, and can be used to recommend items unseen by other users.

The results show that the two components of our system are both performing well. The selection agents are learning increasingly accurate profiles of their users, and the adaptive collection agents are collecting increasingly pertinent web pages, consistently rated as better than pages from the competing sources. The two hypothesized behaviors of specialization and serendipity have both been observed—after even a short time agents are converging on topics, and where common interests between users exist the agent profiles are overlapping. Our hope is to use this type of convergence to be able to serve a greater number of users from a fixed pool of agents—our next experiment will investigate the effects of scaling up the number of users.

References

- Allan, J. 1995. Relevance feedback with too much data. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.
- Armstrong, R.; Freitag, D.; Joachims, T.; and Mitchell, T. 1995. WebWatcher: A learning apprentice for the World-Wide Web. In *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogenous, Distributed Resources*.
- Balabanović, M., and Shoham, Y. 1995. Learning information retrieval agents: Experiments with automated web browsing. In *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogenous, Distributed Resources*.
- Buckley, C.; Salton, G.; Allan, J.; and Singhal, A. 1995. Automatic query expansion using SMART: TREC-3. In *Proceedings of the 3rd Text REtrieval Conference*.
- Cox, III, E. P. 1980. The optimal number of response alternatives for a scale: A review. *Journal of Marketing Research* 17:407–422.
- Deerwester, S.; Dumais, S. T.; Furnas, G. W.; Landauer, T. K.; and Harshman, R. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41(6):391–407.
- Foltz, P. W., and Dumais, S. T. 1992. Personalized information delivery: An analysis of information filtering methods. *Communications of the ACM* 35(12):51–60.
- Harman, D. 1995. Overview of the third Text REtrieval Conference (TREC-3). In *Proceedings of the 3rd Text REtrieval Conference*.
- Lang, K. 1995. Newsweeder: Learning to filter net-news. In *Proceedings of the 12th International Conference on Machine Learning*.
- Lesk, M. E., and Salton, G. 1971. Relevance assessments and retrieval system evaluation. In *The Smart System—Experiments in Automatic Document Processing*. Prentice Hall Inc. 506–527.
- Lieberman, H. 1995. Letizia: An agent that assists web browsing. In *International Joint Conference on Artificial Intelligence*.
- Mauldin, M. L., and Leavitt, J. R. 1994. Web-agent related research at the CMT. In *Proceedings of the ACM Special Interest Group on Networked Information Discovery and Retrieval (SIGNIDR-94)*.
- Mitchell, T. 1995. Personal communication.
- Mittelstaedt, R. A. 1971. Semantic properties of selective evaluative adjectives: Other evidence. *Journal of Marketing Research* 8:236–237.
- Pazzani, M.; Muramatsu, J.; and Billsus, D. 1996. Syskill & Webert: Identifying interesting web sites. In *Proceedings of the 13th National Conference on Artificial Intelligence (to appear)*.

- Porter, M. 1980. An algorithm for suffix stripping. *Program* 14(3):130–137.
- Resnick, P.; Iacovou, N.; Suchak, M.; Bergstrom, P.; and Riedl, J. 1994. GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*.
- Rocchio, Jr., J. 1971. Relevance feedback in information retrieval. In *The Smart System—Experiments in Automatic Document Processing*. Prentice Hall Inc. 313–323.
- Rorvig, M. E. 1988. Psychometric measurement and information retrieval. In Williams, M. E., ed., *Annual Review of Information Science and Technology*, volume 23. Elsevier Science Publishers B.V. 157–189.
- Salton, G., and McGill, M. J. 1983. *An Introduction to Modern Information Retrieval*. McGraw-Hill.
- Saracevic, T. 1975. Relevance: A review of and a framework for the thinking on the notion in information science. *Journal of the American Society for Information Science* 26(6):321–343.
- Saracevic, T. 1995. Evaluation of evaluation in information retrieval. In *18th International ACM SIGIR Conference on Research and Development in Information Retrieval*.
- Schütze, H.; Hull, D. A.; and Pedersen, J. O. 1995. A comparison of classifiers and document representations for the routing problem. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.
- Shardanand, U., and Maes, P. 1995. Social information filtering: Algorithms for automating “word of mouth”. In *Conference on Human Factors in Computing Systems (CHI ’95)*.
- Sheth, B., and Maes, P. 1993. Evolving agents for personalized information filtering. In *Proceedings of the 9th IEEE Conference on Artificial Intelligence for Applications*.
- Yao, Y. Y. 1995. Measuring retrieval effectiveness based on user preference of documents. *Journal of the American Society for Information Science* 46(2):133–145.