User Modeling and User-Adapted Interaction **8:** 71–102, 1998.
© 1998 *Kluwer Academic Publishers. Printed in the Netherlands.*

Exploring versus Exploiting when Learning User Models for Text Recommendation

MARKO BALABANOVIĆ[⋆]
Department of Computer Science, Stanford University, Stanford CA, USA.
e-mail: marko@cs.stanford.edu

(Received 30 September 1997; accepted in revised form 9 February 1998)

Abstract. The text recommendation task involves delivering sets of documents to users on the basis of user models. These models are improved over time, given feedback on the delivered documents. When selecting documents to recommend, a system faces an instance of the exploration/exploitation tradeoff: whether to deliver documents about which there is little certainty, or those which are known to match the user model learned so far. In this paper, a simulation is constructed to investigate the effects of this tradeoff on the rate of learning user models, and the resulting compositions of the sets of recommended documents, in particular World-Wide Web pages. Document selection strategies are developed which correspond to different points along the tradeoff. Using an exploitative strategy, our results show that simple preference functions can successfully be learned using a vector-space representation of a user model in conjunction with a gradient descent algorithm, but that increasingly complex preference functions lead to a slowing down of the learning process. Exploratory strategies are shown to increase the rate of user model acquisition at the expense of presenting users with suboptimal recommendations; in addition they adapt to user preference changes more rapidly than exploitative strategies. These simulated tests suggest an implementation for a simple control that is exposed to users, allowing them to vary a system's document selection behavior depending on individual circumstances.

Key words: Recommender systems, Information filtering, User modeling, Relevance feedback, Selective Dissemination of Information, Machine learning, Adaptive information retrieval

1. Introduction

A variety of systems improve their interactions with users by constructing user models: tutoring systems try to evaluate students' skill levels, on-line help systems aim to determine users' plans, and intelligent assistants provide customized short-cuts for repetitive tasks. *Recommender systems* (Resnick and Varian, 1997) help users keep up-to-date with ongoing areas of interest by regularly recommending a small number of items. These items could be movies, music CDs, news articles, or, as is the focus of this research, World-Wide Web pages. In order to make recommendations appropriate to a user, a model of his or her interests is constructed in order to select a suitable set of Web pages.

[⋆] Gates Building 1A, Stanford CA 94305-9010, USA; marko@cs.stanford.edu

We assume that the user's input to a recommender system consists solely of feedback on the items recommended. As opposed to active participants in a search process, we see users as passive viewers who will supply explicit feedback only grudgingly; they are not willing to navigate large results lists, but must be presented with small sets of recommendations. This simple interaction schema suggests a correspondingly simple user model representation. Unlike richly structured formalisms such as stereotypes (Rich, 1979; Brajnik et al., 1987), in this research we will not make any attempts to ascertain users' age, education level, religious background or other demographic factors. Instead we represent a user by a model of their documents of interest.

An approach to constructing such user models often taken by vendors of online personalized newspapers is to require users to explicitly specify their interests, either by using some query language or by choosing among predetermined categories. The underlying technologies are in fact the same information retrieval tools originally developed to serve short-term information needs. For longer-term, ongoing and potentially changing information needs, specifying precise queries or making a good choice among the supplied categories is difficult. An alternative which places less burden on a user is to use machine learning techniques to automatically generate a user model that can then be used to deliver appropriate recommendations. Therefore we define the *text recommendation* task by the following simple loop:

(1) Select a set of documents to present to the user, based on an induced user model;
(2) Elicit feedback from the user;
(3) Improve the user model given the feedback on the presented documents.

The user models learned, according to the dimensions defined by Rich (1979), are individual user, long-term, and implicitly acquired. According to the classification of Carbonell (1983), they are analytical cognitive models.

Studies of human users provide the most direct means of evaluating recommender systems, but are expensive and time-consuming to perform, and so are often carried out on a relatively small scale, e.g., (Sheth and Maes, 1993; Resnick et al., 1994; Lang, 1995; Pazzani et al., 1996; Balabanović and Shoham, 1997). Instead we construct a simulation. This provides a complementary arena for research: populations of hundreds of users over periods of several virtual months are easier to study, and underlying phenomena can be isolated for more thorough analysis.

One of the challenges in creating such a simulated system is retaining the correspondence to real-world situations. We have endeavored to gather an authentically noisy and irregular corpus of Web pages. However, rather than attempting to simulate the full gamut of human information seeking behavior, we concentrate on a specific task best described as 'following threads of interest' (Jennings and Higuchi, 1993), where specific but ongoing information needs exist. Given such a task, we make a further assumption when simulating users' feedback on recommended pages: we consider only the subject matter of a page, and its relation to a simplified simulation of user interests based on preferences among types of subject matter. Later sections will examine these assumptions and the resulting simulation implementation in more detail.

Unlike information retrieval (IR) or routing systems, a text recommender system selects its own training data, engaging in *active* learning and therefore exposing itself to the *exploration/exploitation tradeoff*. In the field of machine learning, this tradeoff is most commonly discussed in the context of reinforcement learning. The 'k-armed bandit' provides the canonical example: an agent is in a room with k one-armed bandits (gambling machines where there is a probability of winning a prize each time their 'arms' are pulled). On each turn, the agent can pull any arm, with no cost but missed opportunity. If the agent believes one arm as a high payoff, should it choose that arm all of the time (exploitation), or choose another for which it has less information (exploration)? Equating user satisfaction with payoff reveals this same tradeoff in the text recommendation task: should the system recommend pages which it believes the user likes, or recommend pages about which it has little information? A further complication is that a recommender system has a goal which is often in conflict with user satisfaction: that of learning a user model. By performing less exploration, always conservatively recommending documents which are similar to those for which the user has already expressed a preference, the system could be prolonging the time required to learn an accurate user model. Rather than advocating a particular point on this exploration/exploitation tradeoff, or even suggesting that an appropriate balance can be learned by the system, we propose that the user be left in control. The interface to our 'live' recommendation system (whose architecture is described in Balabanović and Shoham, 1997) includes a slider allowing the user to move between different points of this tradeoff.

The purpose of the simulation experiments in this paper is to investigate implementations of such an exploration/exploitation parameter in a controlled environment, and in particular to study its effects on the learning rate of a recommender system and the composition of sets of recommended documents.

The remainder of this paper is structured as follows. In the next section the simulation environment is described in detail, including the representation schemes for users and documents, and the learning algorithm. The experiments are then introduced with a description of the methodology and data used. Finally, the experimental results are analyzed, followed by a discussion of related work and a summary.

2. Simulation Environment

Figure 1 illustrates the main components of the simulation environment. In this paper we consider the case where the recommender system is wrapped inside a simulation system, with a static document collection and simulated users standing

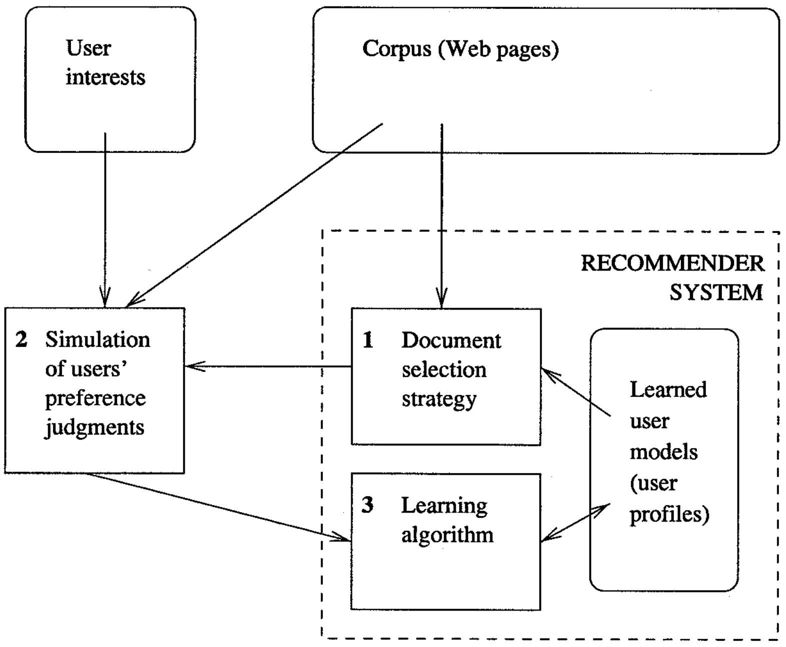
User interests
Corpus (Web pages)
RECOMMENDER SYSTEM
2 Simulation of users' preference judgments
1 Document selection strategy
3 Learning algorithm
Learned user models (user profiles)

Figure 1. Major storage and processing components of the simulation environment, and the data flow between them.

in for their real-world counterparts. It could equally be deployed in a live setting, fetching documents directly from the Web and being accessed by real users.

For a particular experiment, a set of simulated users and their interests are initialized, and a suitable corpus of documents chosen. For each user, each loop of the simulation then proceeds in the following three steps, corresponding to the text recommendation task defined in Section 1 and to components 1, 2, and 3 of Figure 1:

(1) The recommender system selects a set of documents to present to the user, based the user model learned so far;
(2) User feedback on these documents is simulated;
(3) The recommender system then improves the user model given the feedback on the presented documents.

In the remainder of this section we will explain in more detail the functioning of each of the components of Figure 1. Before doing so it will be helpful to introduce some notation. Let $D = \{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n\}$ denote a set of documents. A *preference ranking* $\succ$ is a binary relation on this set:

$$\text{for } \mathbf{d}, \mathbf{d}' \in D, \mathbf{d} \succ \mathbf{d}' \Leftrightarrow \mathbf{d} \text{ is preferred to } \mathbf{d}'. \tag{1}$$

It is possible for two documents to be tied with respect to a preference ranking. We say a ranking $\succ$ is *indifferent* about $\mathbf{d}$ and $\mathbf{d}'$ (written $\mathbf{d} \sim \mathbf{d}'$) if neither $\mathbf{d} \succ \mathbf{d}'$ nor $\mathbf{d}' \succ \mathbf{d}$ holds. This can be interpreted as meaning that both documents are equally preferable, or simply that they are not comparable.

We will assume preference rankings are *weak orders*, in other words that they satisfy the following two axioms, for all $\mathbf{d}$, $\mathbf{d}'$, $\mathbf{d}''$ in D:

$$\text{Asymmetry: } \mathbf{d} \succ \mathbf{d}' \Rightarrow \neg(\mathbf{d}' \succ \mathbf{d}) \tag{2}$$

$$\text{Negative Transitivity: } \neg(\mathbf{d} \succ \mathbf{d}') \wedge \neg(\mathbf{d}' \succ \mathbf{d}'') \Rightarrow \neg(\mathbf{d} \succ \mathbf{d}''). \tag{3}$$

Note that these axioms imply the positive transitivity of $\succ$, and also that $\sim$ is an equivalence relation (Yao, 1995). A weak order can be visualized as a number of levels into which documents are arranged-documents in a higher level are preferred to documents in a lower level, and documents at the same level are indifferent.

2.1. SIMULATING USER PREFERENCE JUDGMENTS

Figure 1 includes the three requirements for simulating user preference judgments of documents: a representation for documents in a corpus; a representation for user interests; and a way of determining users' preferences for a document given their interests.

2.1.1. *Representation of documents*

The basis for the representation of documents in the corpus is the notion of *topical relevance*, which in turn depends on the concept of information need. Taylor (1962) provides a useful breakdown of information need, positing that it progresses through four stages: visceral, conscious, formalized and compromised. The topical relevance of a document is defined as its relevance to a formalized information need, as determined by a human judge. This is the definition of relevance used for evaluating systems at the TREC conferences (Harman, 1996). An example of an intensional definition of a formalized information need from TREC-4 is 'What are the prospects of the Quebec separatists achieving independence from the rest of Canada?' Unlike utility, which is a relation between a user and a document, topical relevance is a relation between a 'topic' and a document, and therefore a less subjective notion.

Editorial categories are defined as clusters of topical relevance to which human editors can assign documents according to their content, and which have been chosen by human editors so as to be useful for particular audiences. Commonly found examples are categories the news services assign to their news stories (e.g., the Reuters service assigns labels such as 'gold', 'cocoa' and 'money supply'), or newspaper sections like 'politics', 'sport', 'financial news', etc. An example of an editorial category analogous to the aforementioned formalized information need is

‘Canadian Current Events: National Unity’, a category used for classifying news articles by the Yahoo!$^{\text{TM}}$ Web service*. One way to think about editorial categories is as extensional definitions of long-term information needs, as defined by editors rather than end-users.

The corpus can now be defined: it consists of a set of editorial categories, each one of which contains a number of Web pages. We use *topic* as a shorthand for editorial category in the remainder of this paper, and use T_i (where T_i is a subset of D, the set of all documents in the corpus) to denote a topic. For simplicity’s sake we assume each page belongs to a single topic. This is not necessary for the functioning of the algorithms but makes for unambiguous experimental results.

2.1.2. *Representation of users’ interests*

Users are assumed to have a preference ranking among the set of topics present in the corpus. If one topic is preferred to another, then all documents from the first will be preferred to all documents from the second. In our experiments users are modeled by a particular class of preference structures. Given two topics T_1, T_2, we write $T_1 \succ T_2$ as a shorthand for $\mathbf{d} \succ \mathbf{d}'$ for all $\mathbf{d} \in T_1, \mathbf{d}' \in T_2$. We define an n-**pref user** as having a preference relation $\succ$ with the following form for topics $T_1, T_2, \ldots, T_m$:

$$T_1 \succ T_2 \succ \cdots \succ T_n \succ \{T_{n+1}, T_{n+2}, \ldots, T_m\}. \tag{4}$$

Thus the preference relation for an n-pref user has $n + 1$ equivalence classes; a 1-pref user has a single topic which is preferred to all others, a 2-pref user has two topics they prefer to all others, and in addition has a preference among those two, and so on.

This n-pref construction relies on four key assumptions:

(1) Topical relevance is the only factor influencing users’ judgments of documents that is considered;
(2) Topical relevance is expressed in terms of editorial categories, or topics;
(3) A user’s interests are simulated by a preference ranking among these topics;
(4) Such a preference ranking is of the form of an n-pref structure, as defined above.

Preference rankings over editorial categories are not a user model representation commonly seen in the literature. Therefore the remainder of this section will examine the reasoning behind each assumption in turn.

Users cite a wide variety of reasons for their preferences among documents. Many of the pertinent studies agree, however, that topical relevance is a crucial factor influencing user judgments. Detailed descriptions of the various methodologies are beyond the scope of this paper, but to give a flavor for the results: In a study

* The Yahoo! hierarchy is available at http://www.yahoo.com.

where people were asked for criteria they used in evaluating the value of a document in relation to a specific information need, 82% of them cited topical relevance as a primary reason (Schamber and Bateman, 1996). In another study, analysis of think-aloud protocols from an information seeking task revealed topical relevance to be the major factor (Wang and Soergel, 1996). A related information seeking task is selecting articles to read from an electronic newsgroup. In a study of why people chose to read particular Usenet news articles, topical relevance was shown to play a part in almost all of the decisions (Stadnyk and Kass, 1991). This study is more relevant to the text recommendation task than those previously mentioned, as it concerns long-term rather than short-term information needs.

All of the studies cited above also give *novelty* as an important factor, often second only to topical relevance. Novelty here is loosely defined as the existence of information in a document which is new to a user. The *probability ranking principle* (Robertson, 1977) states that the optimal output for an IR system is a list of documents ranked in order of estimated probability of relevance to the supplied query. Since most IR systems adhere to this principle and evaluate each document independently, they are ill-equipped to model set-based factors like novelty (topical relevance of a document can be determined solely from the document itself, whereas novelty requires analysis of the preceding set of documents). In practice, IR systems (e.g., some Web search engines) take the first steps towards addressing this factor by removing duplicates from a results list. Often there is a tradeoff between topical relevance and novelty – the most topically relevant documents for some information need would usually also be very similar to each other. Without further empirical investigations it is unclear how successfully IR similarity measures, for instance, would model human novelty judgments. Novelty, therefore, will not form a part of the experiments we perform.

A variety of further criteria have been identified which can be thought of as inherent in the content of the document. These include the ‘type’ of the document (e.g., ‘request for information’, ‘advertisement’, ‘discussion’, etc.), the source of the document, the quality of writing, the ‘sensationalism’ of the document. In addition, a large number of factors dependent upon the user’s context have been identified in many studies, and each user will also bring their own set of criteria to bear. Commonly cited factors include: knowledge of domain, time constraints, emotional appeal, societal importance, media cueing, credibility, popularity. We will not attempt to simulate these additional factors, and indeed for many of them this would be difficult or impossible. Most interactive information systems present affordances allowing users to apply their own idiosyncratic criteria by manipulating or choosing among the set of retrieved documents.

The choice of editorial categories as indicators of user interest is partially a pragmatic one – a corpus can automatically be gathered, and simulated user interests can be generated without human involvement. Nevertheless, empirical evidence exists associating such categories with long-term, ongoing information needs. In studies of newspaper readers, the distribution of editorial categories among articles selected for reading has been found to be predictable and stable over periods of several weeks (Allen, 1990; Graber, 1988, Chapter 5), despite the difficulty of predicting choice among individual articles. Although newspaper articles can be thought of as a subdomain of the Web, we anticipate that this behavior would still be exhibited.

It is common in studies such as this to use simple binary relevant/irrelevant classifications when modeling users' judgments of topical relevance. There is significant evidence, however, which suggests that these judgments are more faithfully modeled by a preference ranking. Rorvig (1988) cites experiments which show that human relevance judgments can be expressed on a transitive, interval scale (by relating them to judgments of 'prothetic' stimuli such as loudness, weight, etc.); the consistency of relative relevance judgments compared to absolute ones is shown in several empirical studies (Lesk and Salton, 1971; Saracevic, 1975). Furthermore, absolute relevance judgments can themselves be expressed using appropriately chosen preference rankings. For instance, a 1-pref user induces a binary classification, allowing comparison with techniques which make the prevalent assumption of binary-valued relevance judgments.

In order to actually run a simulation, it is necessary to pick particular preference rankings to represent user interests. The n-pref structure is a simple example which allows variation in the complexity of preferences. The condition that preferences among individual documents always follow the preferences among the corresponding topics may at first seem unrealistically strict. However, as will be seen, the corpus used for simulations contains topics which are relatively broad and messy; in addition, the constituent documents are often irregular, and contain various multimedia or interactive components which are difficult for a computer system to interpret. In effect, the unfocused nature of the topics balances out the strict nature of the preference. Furthermore, as explained above, the n-pref structure is actually more expressive than the binary relevance classifications normally used when testing information filtering or routing systems (e.g., at the TREC conferences).

Interestingly, in early tests we observed that replacing a topic with the union of several topics had little effect on the learning rate. For instance, we simulated a 1-pref user where topic T_1 actually corresponded to the union of two editorial categories, rather than just a single category: results were very similar to regular 1-pref user results. This robustness to variation in the composition of topics suggests that different preference structures using the same $\succ$ relation would not shed any new light on the specific issues investigated by our experiments.

2.1.3. *Simulating preferences among documents*

Given a preference ranking over editorial categories, the simulation system can simulate a user's topical relevance judgments to supply an ordering for a pair of documents – it will equate to the ordering for the corresponding editorial categories, if one exists, otherwise the documents will be regarded as equivalent.

A problem for a live recommender system which does not occur in this simulated setting is the elicitation of such preference rankings from users. An implicit way to accomplish this is to monitor document selection behavior (Cohen et al., 1997). For instance, the user can be presented with a list of recommendations, sorted according to a ranking predicted from a user model. If the user elects to read the third document first, it can be inferred that the third document should have been ranked first. Further feedback can be obtained in certain settings, for instance if users' Web browsing or bookmarking can be monitored (Rucker and Polanco, 1997). By not requiring a user to disengage from the task at hand in order to decide upon and input explicit feedback, a recommender system stands a greater chance of being used more regularly, leading to more frequent feedback, more accurate user models and ultimately better recommendations.

2.2. RECOMMENDING DOCUMENTS

This section will describe the recommender system (shown in Figure 1), which learns models of the simulated users and selects sets of documents to recommend.

2.2.1. *Parsing of documents*

A prerequisite for the application of machine learning techniques to the Web pages of the corpus is the extraction of features. We use the standard vector-space model of documents (Salton and McGill, 1983), where each document corresponds to a vector in a space whose dimensions correspond to different words from some dictionary. In the set of documents $D = \{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n\}$, each $\mathbf{d}_i$ is a vector in a p-dimensional vector space over a set of keywords $\{t_1, t_2, \ldots, t_p\}$.

The major assumption of this representation is that the content of a Web page can be represented purely by the set of words contained in the text. All HTML tags, comments, images or other multimedia objects are ignored. In addition very common English words from a standard stop list of 571 words are ignored (some of the parsing methods employed are English language-specific, given our English-speaking target audience).

Extracted words are reduced to their stems using the Porter algorithm (Porter, 1980), and then weighted using a 'TFIDF' scheme: the weight d_{t_i} of a word t_i in a document $\mathbf{d}$ is derived by multiplying a term frequency ('TF') component by an inverse document frequency ('IDF') component:

$$d_{t_i} = \left(0.5 + 0.5\frac{tf(i)}{tf_{\max}}\right)\left(\log\frac{n}{\mathrm{d}f(i)}\right), \tag{5}$$

where $tf(i)$ is the number of times word t_i appears in document $\mathbf{d}$, $\mathrm{d}f(i)$ is the number of documents in the corpus which contain t_i (the *document frequency*), n is the number of documents in the corpus and $tf_{\max}$ is the maximum term frequency over all words in $\mathbf{d}$. If word t_i does not occur in document $\mathbf{d}$ then $\mathbf{d}_{t_i} = 0$. In a final step, the vector for each document is normalized to be of unit length. Variants of this formula are commonly used by IR systems – the aim is to give high weights to words which occur frequently in the document in question, but are rare in the rest of the corpus.

As a text recommender system only sees documents in small batches, rather than having access to a large corpus all at once, it is impossible to accurately measure the document frequencies of words as required by the TFIDF formula. Similarly, the feature selection schemes commonly used in text categorization experiments (often based on information theoretic measures, e.g., Koller and Sahami, 1996) are not appropriate, since they also assume a batch rather than incremental model of learning. Instead, the document frequencies are roughly 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 $n = 5,229$). Thus in a live recommender system the representation for a document could be computed immediately upon the document's discovery.

2.2.2. *Representation of user profiles*

The recommender system represents each user with a form of user model which we call a *user profile*. For our purposes, a user profile needs to be capable of representing an arbitrary set of preferences among documents. Note that since the recommender system only receives feedback at the document level, it neither requires nor depends on the notion of editorial categories employed by the simulation system.

In this section we give an overview of an adaptive linear system (Bollman and Wong, 1987; Wong et al., 1988), which forms the basis for our user profile representation and learning algorithm.

If D is a countable set, then it has been shown (Roberts, 1979) that there exists a function $f\colon D \mapsto \mathbb{R}$ such that:

$$\mathbf{d} \succ \mathbf{d}' \Leftrightarrow f(\mathbf{d}) > f(\mathbf{d}') \quad \text{for all} \quad \mathbf{d}, \mathbf{d}' \in D. \tag{6}$$

We represent each user profile as a vector $\mathbf{q} = [w_{t_1} w_{t_2}, \ldots, w_{t_p}]$ in the same p-dimensional vector space as the documents, so each element w_{t_i} is a weight corresponding to word t_i. We denote the transpose of $\mathbf{q}$ by $\mathbf{q}^T$. Bollman and Wong (1987) show various conditions for different information retrieval models under which f can be a linear function:

$$f(\mathbf{d}) = \mathbf{q}^T \mathbf{d} = \sum_{1 \leqslant i \leqslant p} w_{t_i} d_{t_i}. \tag{7}$$

We say a preference relation $\succ$ is *weakly linear* (Wong and Yao, 1990) if there exists a user profile $\mathbf{q}$ such that for any $\mathbf{d}, \mathbf{d}'$ in D,

$$\mathbf{d} \succ \mathbf{d}' \Rightarrow \mathbf{q}^T \mathbf{d} > \mathbf{q}^T \mathbf{d}'. \tag{8}$$

In our experiments user profiles are represented using vectors $\mathbf{q}$ as defined above. Clearly if a ranking $\succ$ is not weakly linear, then a user profile inducing it cannot be expressed using our feature set and vector representation. In practice such an exact solution is unrealistic, and we seek to attain a good approximation. The extent to which our corpus and feature set meet the weak linearity condition will be determined empirically.

One known deficiency of representing a user profile as a single vector is the inability to represent the EXCLUSIVE-OR relation among words. For instance, a user interested in both hiking (trekking) and astronomy (stars) might be recommended unwanted documents about the 'Star Trek' television show. In practice, however, the vocabulary is usually rich enough to ameliorate such effects.

The *score* for a document $\mathbf{d}$ relative to a profile $\mathbf{q}$ is defined to be $\mathbf{q}^T\mathbf{d}$. To create a predicted ranking over a set of documents based on a profile, the documents are simply sorted according to their score.

2.2.3. *Learning algorithm*

If the user does have a weakly linear preference function, which can be represented faithfully by a vector $\mathbf{q}$, then $\mathbf{q}$ can be found using a simple gradient descent procedure (Wong and Yao, 1990), which is just a variant of the perceptron learning rule (Rosenblatt, 1960).

On each loop of the simulation the user supplies the desired ranking for a set of documents. In addition, the recommender system can use the profile learned thus far to predict a ranking over the same set. An error vector can then be computed for each pair of documents that the recommender system ranked incorrectly. If the user is indifferent as to the ranking of a pair of documents, then the recommender system is permitted to rank them in any order. This is known as the acceptable ranking criterion (Wong et al., 1988) and is commonly used in IR evaluations. An error vector is just the difference between two incorrectly ranked document vectors. These error vectors are added to the existing profile vector to update it for the next step.

Formally, if $\mathbf{q}_k$ is the profile after the kth step, then:

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \sum_{\mathbf{b}\in\Gamma(\mathbf{q}_k)} \mathbf{b}, \tag{9}$$

$$\text{where} \quad \Gamma(\mathbf{q}_k) = \{\mathbf{b} = \mathbf{d} - \mathbf{d}' | \mathbf{d} \succ \mathbf{d}', \mathbf{q}_k^T\mathbf{b} \leqslant 0\}.$$

A single step of this procedure is very similar to relevance feedback (Rocchio, 1971), generating Rocchio's *optimal query* if started with an empty vector. In a simple case where preferences consist of just two equivalence classes, relevant and irrelevant, the gradient descent procedure has been empirically demonstrated to have comparable precision/recall results to commonly used relevance feedback methods (Wong et al., 1991). Results shown in Section 4.4 confirm this for our corpus, although regular relevance feedback is not applicable for cases where the preference structure is more complex. Many other machine learning algorithms could have been considered, for instance using more expressive formulations such as multilayer neural networks. However an advantage of the scheme described is that it is not impeded by the high dimensionality of the space of documents. Approaches taken by other researchers are discussed in Section 5.

An important choice when implementing this algorithm is the value of the initial vector $\mathbf{q}_0$. Given a weakly linear preference function, the procedure is guaranteed to converge after a finite number of steps, regardless of the value of $\mathbf{q}_0$ (Wong and Yao, 1990) (in other words, there will be a step k where $\Gamma(\mathbf{q}_k) = \emptyset$). However, there may be a large number of possible solution vectors given the training data. As for any inductive algorithm, there is no guarantee that any of these will be solution vectors over unseen documents. It has been shown that a well-chosen starting vector can considerably improve performance (Wong and Yao, 1990), and we expect this effect to be accentuated if the procedure is halted before convergence has occurred.

So far, the gradient descent procedure has been described with respect to a single set of documents where the desired ranking is known. In practice, the recommender system must update a user profile every time it receives preference information over the set of documents delivered to the user on the previous iteration.

Two different schemes can be imagined:

Single Step. In the simplest scheme, given a previously computed user profile and a set of recommendations with associated preference information, the resulting errors can be added to the existing profile just once, i.e., taking one step of gradient descent. Therefore after any set of recommendations, the learning system has effectively performed a single step of gradient descent over all of the documents recommended thus far. The advantage of this method is that previously recommended documents and preferences among them need not be stored. Storage is only required for a user profile per user. The disadvantage is that only a single step of gradient descent can be performed.

Multiple Steps. In a more complete scheme, preferences and document representations for all recommendations ever provided are stored. User profiles can be computed afresh at any time, over all the documents recommended thus far. Any number of gradient descent steps can be performed. In a modification of this scheme, a previously computed user profile can be used as a starting point for the gradient descent procedure. Note that by storing preferences for only a fixed amount of time, or by decreasing the influence of older preferences, it is easy to implement a bias for newer judgments using this scheme (for the single step scheme, it would be necessary to decay weights in the learned profile).

The relative performance of these two schemes is investigated in the experiments to follow.

2.2.4. *Document selection strategies*

Imagine a scenario where a user is interested in regularly receiving both documents about music and documents about literature. By random chance a recommender system might deliver a document about music in its first set of recommendations. If the user ranks this document highly, the system might then be so successful in learning this single topic preference that all documents subsequently delivered are on the topic of music. This problem of over-specialization is well known in information filtering or recommender systems (Sheth and Maes, 1993). Such a system is pursuing a purely exploitative document selection strategy, and by restricting its own training data is not learning about other interests the user might have. Our hypothesis is that where a user has more than a single topic of interest, a more exploratory document selection strategy will lead to a faster learning rate for the user profile. However, a *purely* exploratory strategy would never recommend documents to the user similar to those preferred in the past, which seems hardly likely to please the average user.

In order to investigate this instance of the exploration/exploitation tradeoff, it is necessary to define the various document selection strategies more precisely. For each of the following strategies, we only recommend to a user documents they have not already seen. This trivial application of the novelty factor over topical relevance merely prevents endless loops in the simulation; in more realistic systems a more sophisticated rule would be required. For instance, on the Web there are instances of different versions or formats of the same document, ‘mirrored’ versions at different locations, dynamic or changing documents, where the change could be in the content or, for instance, just a new advertisement supplied with every viewing, and so on.

Pure Exploitation
On each iteration, the system retrieves those documents which score highest against the current user profile. Unless otherwise specified, all of the experiments described use this document selection strategy, which is the norm for IR systems.

Pure Exploration
For each user, a record of the words encountered in recommended documents is maintained. In practice this is easy to include with the representation of the user profile. Now, when the user profile is compared to a document, two different values can be derived. The first is the score, as defined in Section 2.2.2. The second we call the *overlap*, and it is the number of terms in common between those the user has seen and those contained in the document. The overlap can be thought of as a measure of the certainty of the value of the score. For instance, if the score value is derived from an overlap of a single word, it is probably less reliable than one derived from an overlap of 60 words.

Note that in a regular IR setting this measure does not make sense because of the implicit closed-world assumption – words which do not appear in a query are assumed to be words which are of no interest to the user. However, when a user profile is learned over time, missing words are merely words about which there has been neither positive nor negative feedback yet.

Thus for the pure exploration strategy, documents with minimal overlap are retrieved (ties are broken randomly). These are documents where the recommender system has the least information about user preference.

Mix of Exploitation and Exploration
This strategy simply involves mixing the top documents recommended by the pure exploitation and pure exploration strategies, in varying proportions.

3. Experimental Setup

To run an experiment using the described simulation environment, an appropriate corpus must be chosen and a particular set of user preferences generated; in addition a method must be decided upon for measuring the performance of the recommender system.

3.1. CORPUS

All of the experiments described were performed on a collection of Web pages gathered by following links from the Yahoo! topic hierarchy. This is a collection of Web pages organized by human editors into a hierarchy according to topical relevance, and so fits the notion of editorial categories.

Ten topics were gathered (Table I) during May and August 1997, a total of 7,840 pages. For each topic, a 1-ply search was conducted following all hyperlinks from the starting URL, so in essence each gathered topic includes two levels of the hierarchy. Links which pointed 'up' the hierarchy or to generic Yahoo! help information were ignored. In addition, pages which occurred in more than one topic were removed (a total of 23 pages, or about 0.3% of the total). The robot exclusion standard (Koster, 1994) was respected – portions of sites which barred access to automatic indexers or agents were ignored. Finally, links to internet domains of non-English speaking countries (e.g., `.fr` for France) were not followed.

Each gathered topic was randomly split, with 75% of the documents going to a training set and the remaining 25% to a separate test set (in total the training set had 5,884 pages and the test set 1956). We wished to retain the noisiness characteristic of Web data, and so the corpus was left as near as possible to its raw state. As a result, many documents contained few or no words (e.g., 676 documents, about 9% of the total, had vectors with fewer than 10 words, and 45 documents had empty vectors).

Table I. Topics used for test corpus.

Name of topic	URL (http://www.yahoo.com prefix omitted)	Number of pages
Music	/Entertainment/Music/	788
Travel	/Recreation/Travel/	1463
Religion	/Society_and_Culture/Religion/	790
Finance and Investment	/Business_and_Economy/Finance_and_Investment/	518
Multimedia	/Computers_and_Internet/Multimedia/	512
Outdoors	/Recreation/Outdoors/	655
Auto Racing	/Recreation/Sports/Auto_Racing/	490
Literature	/Arts/Humanities/Literature/	1470
Food and Eating	/Entertainment/Food_and_Eating/	457
Law	/Government/Law/	697

The categories above are clearly only a small sample of the many possible clusters of topical relevance users might find interesting. However, since each category encompasses a diverse set of Web pages judged by a human editor to be topically similar, we hope that it is also provides an appropriate example with which to test our system, and that the results would apply equally to further editorial categories. It would be interesting to survey the extent to which clusters of topical relevance as defined by end-users exhibit similar properties.

3.2. USERS

Various preference structures from the class of n-pref users are compared in our experiments. Given a particular preference structure, for instance for a 1-pref user $T_1 \succ \{T_2, T_3, \ldots, T_m\}$, topics from the corpus are randomly assigned to labels $T_1, T_2, \ldots, T_m$. In this way different users with the same preference structure are generated. Since there are 10 topics altogether in the Yahoo! corpus, a 9-pref user is the most complex possible in this scheme – there are 10! or 3,628,800 possible different 9-pref users. For practical purposes, all of the experiments described were performed with a sample of 500 randomly generated users, or all of the possible different users if there were less than 500.

3.3. EVALUATION METRIC

The usual IR metrics of precision and recall assume a binary-valued definition of relevance, and so are inappropriate for the case where users have multilevel preferences. Machine learning experiments in this domain often report results in terms of classification accuracy, but again this is only appropriate for binary preference functions. Instead, we shall use a metric which measures the difference between desired and predicted rankings. A decrease in such a measure, over time, demonstrates that increasingly accurate user models are being learned.

The *ndpm* measure of normalized distance-based performance (Yao, 1995) is defined with reference to pairs of documents in rankings. The reader is referred to Yao's article for the full derivation of this measure, and its relationships to other IR measures. Here we briefly summarize the necessary calculation.

If $\succ_p$ is the ranking predicted by the recommender system, and $\succ_d$ is the simulated user's desired ranking, then we define the distance between two documents $\mathbf{d}, \mathbf{d}' \in D$ with respect to these rankings as follows:

$$\delta_{\succ_p,\succ_d}(\mathbf{d},\mathbf{d}') = \begin{cases} 2 & \text{if}(\mathbf{d} \succ_p \mathbf{d}' \text{ and } \mathbf{d}' \succ_d \mathbf{d}) \text{ or } (\mathbf{d} \succ_d \mathbf{d}' \text{ and } \mathbf{d}' \succ_p \mathbf{d}) \\ 1 & \text{if}(\mathbf{d} \succ_d \mathbf{d}' \text{ or } \mathbf{d}' \succ_d \mathbf{d}) \text{ and } \mathbf{d} \sim_p \mathbf{d}' \\ 0 & \text{otherwise}. \end{cases} \tag{10}$$

Note the asymmetry of this calculation introduced by the acceptable ranking criterion: it does not matter how the recommender system ranks documents which the desired ranking has in the same equivalence class. Now the full measure can be defined by summing δ over all unordered pairs of documents in D, and normalizing by dividing by twice the number of pairs in the desired ranking $\succ_d$:

$$ndpm(\succ_p,\succ_d) = \frac{\sum_{\mathbf{d},\mathbf{d}' \in D} \delta_{\succ_p,\succ_d}(\mathbf{d},\mathbf{d}')}{2|\succ_d|}. \tag{11}$$

This definition ensures the *ndpm* value is in the range [0, 1], with a value of 0.5 representing 'baseline' performance – if the predicted ranking consists of a single equivalence class, then *ndpm* will always evaluate to 0.5.

3.4. METHODOLOGY

Each experiment begins with generating preference rankings among topics for each of the simulated users, according to the n-pref scheme already described. Profiles for each user are initialized as empty vectors. The experiment then proceeds in discrete iterations. Each iteration is either a training iteration or a test iteration, as explained below. Four training iterations are followed by a test iteration; test iterations are numbered 0, 5, 10, 15, etc.

Training Iteration. A training iteration is a regular iteration of the simulation system. On each such iteration, for each user, a set of six documents is selected from the unseen portion of the training set, using the current document selection strategy and the user's current profile. The user's ranking of these six documents is simulated according to their known topic preferences. The learning algorithm uses this ranking and the document representations to update the user profile, according to the gradient descent rule.

Test Iteration. On each test iteration, for each user, the entire test set is ranked according to the user profile, forming the predicted ranking. The desired ranking is known from the user's topic preferences. The $ndpm$ difference between these rankings is calculated and recorded.

Most of the graphs to follow plot learning curves showing the value of $ndpm$ at each test step, averaged over all users (namely, Figures 5, 7, 8, 10 and 11). Learning curves are especially appropriate for recommender systems, since speed of adaptation could form a deciding factor in users' decisions as to whether to use a system.

4. Experimental Results

4.1. UPPER BOUND PERFORMANCE

In Section 2.2.2 we stated the weak linearity condition, necessary for learning user profiles which precisely induce the desired ranking. In this section we examine the extent to which this condition holds given our particular document representation scheme and corpus.

We apply the gradient descent procedure directly on the test set to determine an upper bound on the performance of our system. This is usually referred to as the retrospective test (Robertson and Sparck Jones, 1976). Furthermore, the rate of convergence to a correct user profile provides insight into the suitability of the linear model for this dataset (very slow convergence would indicate that perhaps more complex nonlinear functions were necessary).

The gradient descent procedure (Equation 9) identifies on the kth step the number of pairs of documents ranked incorrectly by the user profile of the $(k-1)$th step. The graph of Figure 2 shows how the percentage of incorrectly ranked document pairs (plotted here on a $\log_{10}$ scale) decreases with each step of gradient descent, as the procedure converges. Note that pairs about which the desired ranking is indifferent were ignored, so the percentage represents the ratio between the number of incorrectly ranked pairs and the total number of pairs where a preference was required.

It is clear that the majority of the convergence occurs in the first step. The amount of computation involved in a gradient descent step is approximately proportional to the number of incorrectly ranked pairs. As the user profile converges, therefore, the amount of work for each step becomes correspondingly less. Although this would seem to suggest that it is worthwhile performing extra steps, the significant storage overhead of retaining past recommendations and preferences, required for the multiple steps gradient descent scheme, could outweigh these considerations.

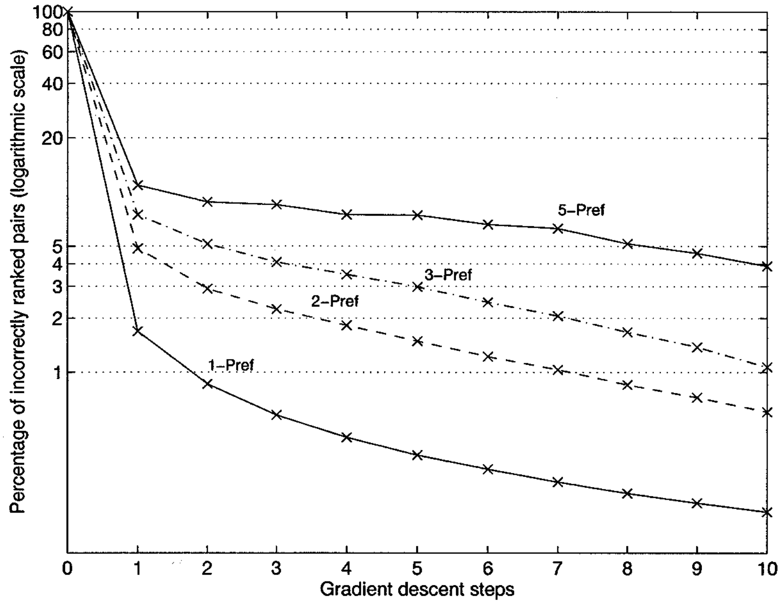

Figure 2. Gradient descent on test set (test of optimal learning), in each case averaged among all possible users with the same preference structure, or 500 randomly generated users, whichever is the lesser.

The results of Figure 2 suggest that for simpler preference structures the adaptive linear model is well suited to this domain – the feature set and user profile representation are sufficiently rich to capture a good approximation of the preferences in a small number of iterations. Looking at the 2-pref users, after 10 steps, on average less than 1% of pairs are incorrectly ranked. However, full convergence has not occurred for any one user, and this does not happen until the 50th step. Given the noisy nature of the dataset, it seems wise to ignore these few stubborn pairs as outliers.

As the preference structures increase in complexity, the optimal performance of the linear model worsens, and it would be interesting in these cases to compare the performance of some of the nonlinear models which have been proposed, e.g., by (Fuhr, 1989).

By taking the user profiles generated by the gradient descent procedure over the test set, values of the *ndpm* measure can also be calculated for various cases. Table II shows the results, which are significantly superior to subsequent runs where the training set is used for learning.

4.2. OPTIMIZING DOCUMENT VECTOR LENGTH

Various previous studies (e.g., Pazzani et al., 1996) have shown that using all of the available words from a document does not necessarily lead to the best learning

Table II. Comparison of *ndpm* values given different numbers of preferences held by users, after 10 steps of gradient descent directly on the test set.

Number of preferences	Average *ndpm* after 10 steps
1	0.0026
2	0.0099
3	0.0224
5	0.0608

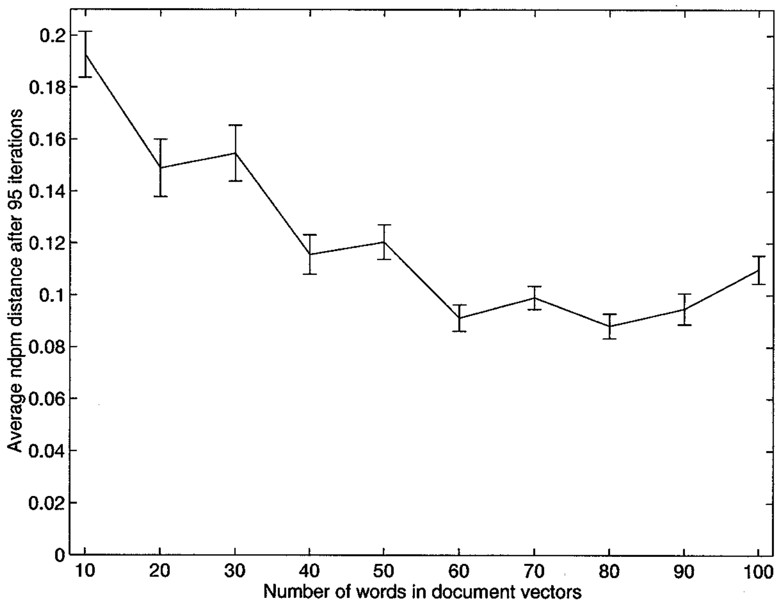

Figure 3. Variation of *ndpm* value after 95 iterations of the recommender system with differing numbers of words used from each document, for 1-pref users. Error bars show 95% confidence intervals.

performance, even apart from the problems standard machine learning algorithms face due to the high-dimensionality of document spaces. This is an instance of the standard problem of over-fitting the training data.

In accordance with standard IR practice, a stop list and stemming are used to reduce the dimensionality of the problem somewhat. Retaining only the top-weighted words from each document allows for even more selectivity.

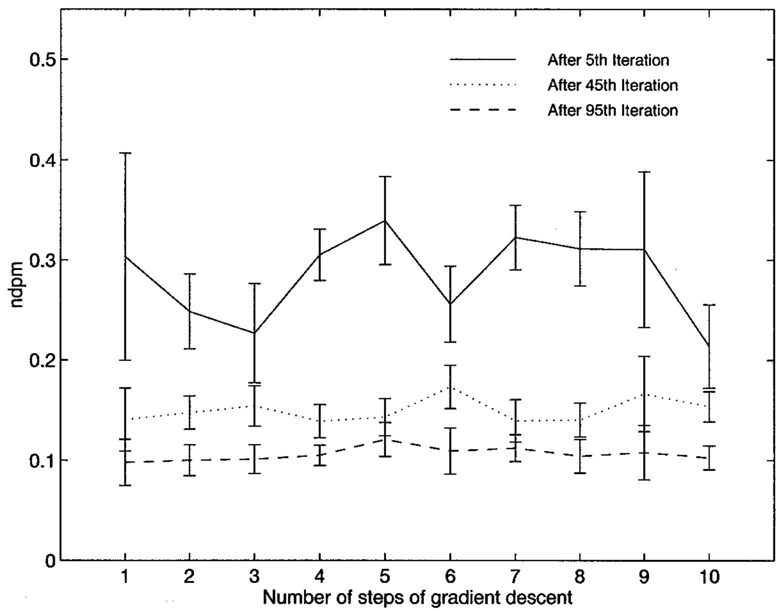

Figure 4. Comparison of *ndpm* distances using different numbers of steps of the gradient descent algorithm, after different numbers of iterations of the recommender system, for 1-pref users. 95% confidence intervals shown.

As opposed to the retrospective tests of the previous section, from here on we follow our stated methodology of alternating four training iterations with a test iteration. Figure 3 shows a series of experiments where the maximum number of words used from each document is varied, from 10 to 100 (in some instances the total number of words in a document will be less than this maximum). The single step gradient descent scheme was followed. The differences in the 40–100 word range are not particularly significant. Further tests showed no improvement when using more than 100 words per document. All of the other experiments described here were performed using only the 60 highest-weighted words from each document.

4.3. COMPARING GRADIENT DESCENT STRATEGIES

Figure 4 shows how performance varies against the number of steps of gradient descent done for each simulation iteration, for 1-pref users. As would be expected from the convergence rate shown in Figure 2, there is little improvement beyond the first step of gradient descent. Therefore the single step scheme is employed for the remainder of the experiments described herein.

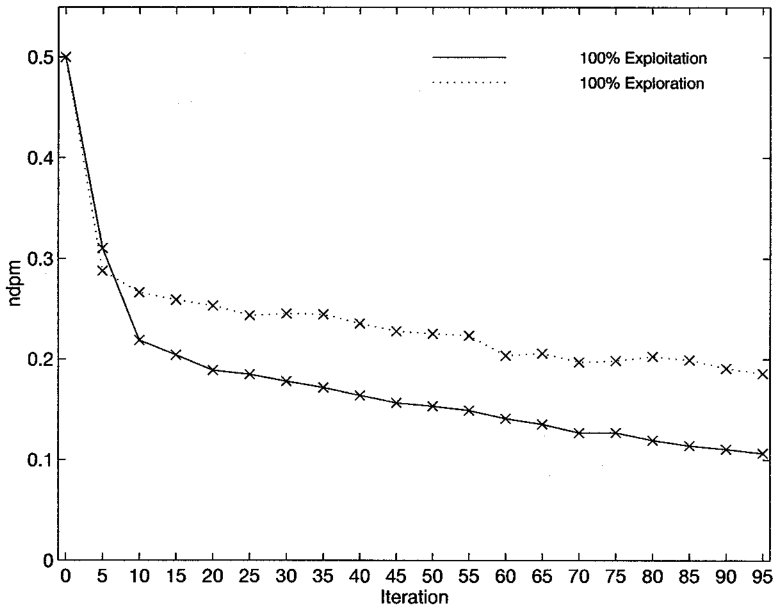

Figure 5. Exploitation vs. Exploration document selection strategies for 1-pref users. Graph shows *ndpm* values averaged over all 10 possible users, measured at test iterations (every fifth iteration).

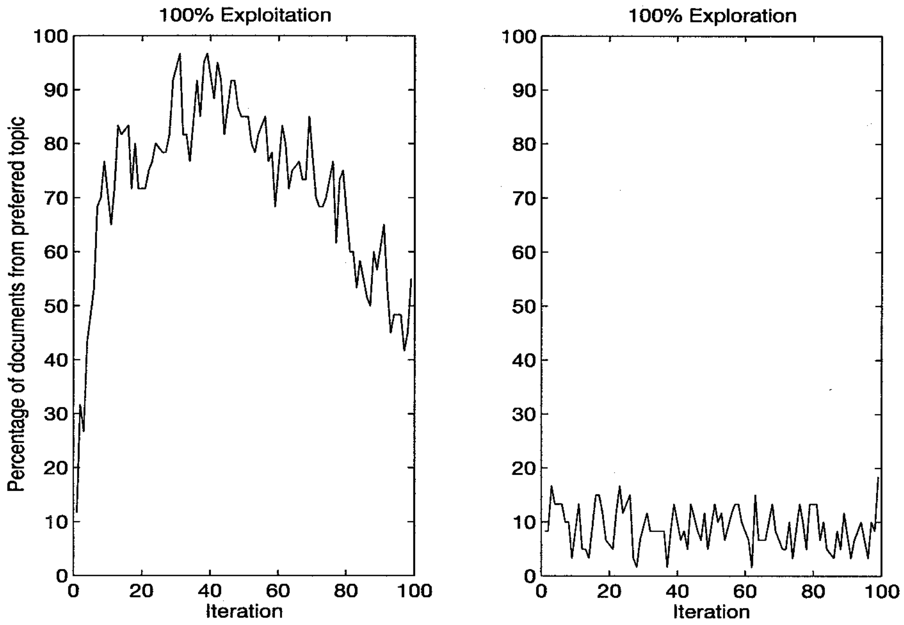

Figure 6. Composition of documents in recommended sets for 1-pref users: 100% exploitation and 100% exploration strategies. Recorded for every training step, and averaged over all 10 possible users.

4.4. COMPARING DOCUMENT SELECTION STRATEGIES

Single preference, or 1-pref, users provide the simplest test case for comparing document selection strategies. Figure 5 shows the learning curves over time for the pure exploitation and pure exploration strategies. As expected, there is no benefit to exploration in this case, since over-specialization to one topic is exactly what is required. Figure 6 shows the composition of recommended documents resulting from the two strategies. As can be seen, a much greater proportion of documents come from the single preferred topic in the exploitation case, and so in all likelihood this would be the optimal strategy if a user really had such a simple preference structure. Note that even for the exploitation strategy, in later iterations the percentage of documents from the preferred topic drops – it becomes more difficult to retrieve relevant documents from the corpus as it is depleted of those in the preferred topic.

In this case only we are able to make a comparison with a classical Rocchio relevance feedback scheme. If $\mathbf{q}_k$ is the user profile after the kth iteration, the relevance feedback profile update rule used can be written:

$$\mathbf{q}_{k+1} = \mathbf{q}_k + 2 \sum_{\mathbf{r} \in R_{k+1}} \mathbf{r} - \frac{1}{2} \sum_{\mathbf{n} \in N_{k+1}} \mathbf{n}, \qquad (12)$$

where, from the set of documents recommended on the $(k+1)$th iteration, R_{k+1} is the subset which are in the preferred topic, and N_{k+1} is the remaining subset which are not in the preferred topic. The same word weighting scheme as already described was used.

The results are shown in Figure 7. Despite not having the extra information inherent in absolute scores, the gradient descent procedure performs comparably on this dataset. Further comparisons would be interesting with machine learning techniques which have been adapted for the TREC routing task, for instance the k-nearest neighbors algorithm (Yang, 1994), although the iterative nature of the text recommendation task makes this difficult.

Learning rates for a population of 3-pref users are shown in Figure 8. As predicted, in this case an exploratory strategy leads to faster user profile acquisition, as a purely exploitative strategy is likely to deliver the majority of documents from a single topic. Indeed, Figure 9 shows that the composition of documents from a purely exploratory strategy is less dominated by documents from the most preferred topic. This provides an excellent illustration of the exploration/exploitation tradeoff: faster learning versus more documents delivered from preferred topics. It can be seen that a strategy of mixing 50% exploitative choices and 50% exploratory choices leads to an intermediary learning rate and an intermediary document composition. Further intermediate mixes that were tried fell between the results shown as would be expected – for clarity they have not been shown in Figures 8 or 9. These results are encouraging – they suggest that varying the composition of recommendation sets by changing the mix of exploratory and exploitative choices would provide the user with a predictable and well-behaved control.

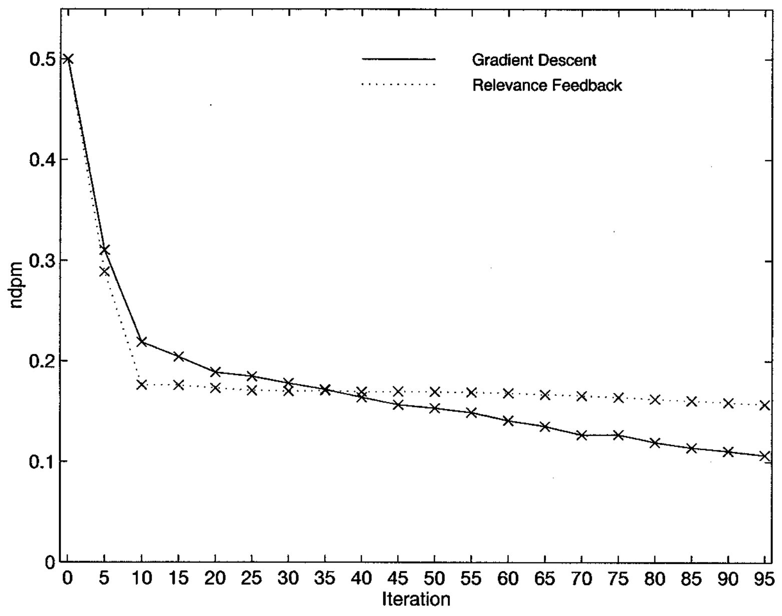

Figure 7. Learning curves for 1-pref users comparing a single iteration of gradient descent to relevance feedback.

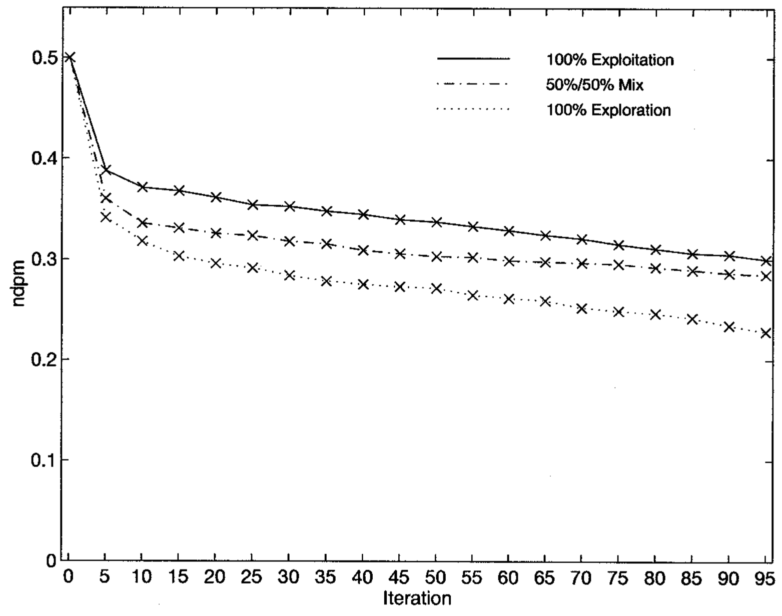

Figure 8. Exploration vs. exploitation document selection strategies for 3-pref users. Graph shows *ndpm* values averaged over 500 randomly selected users, measured at test steps every fifth iteration.

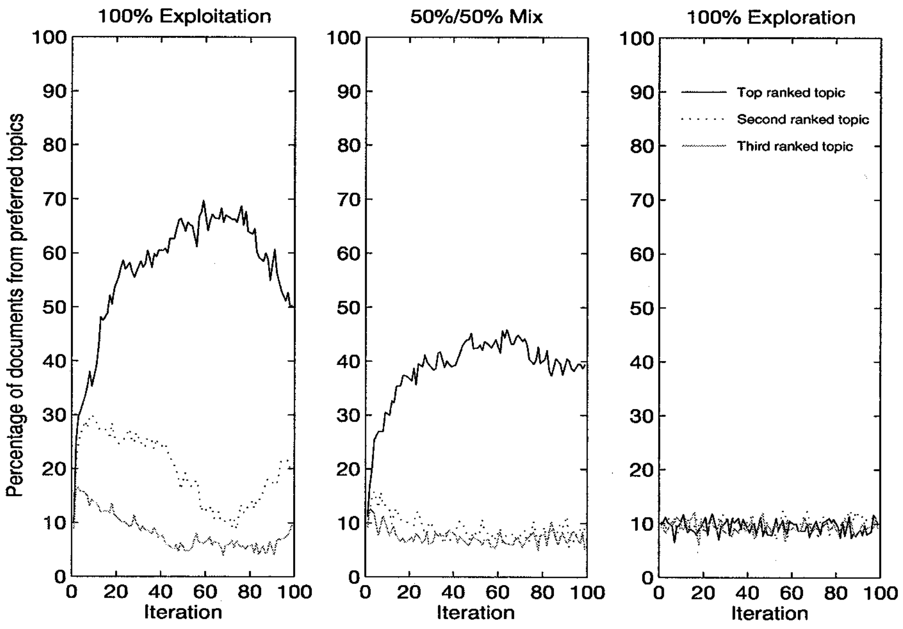

Figure 9. Composition of documents in recommended sets for 3-pref users: 100% exploitation, 50%/50% mix and 100% exploration strategies. Recorded for every training step, and averaged over 500 randomly selected users.

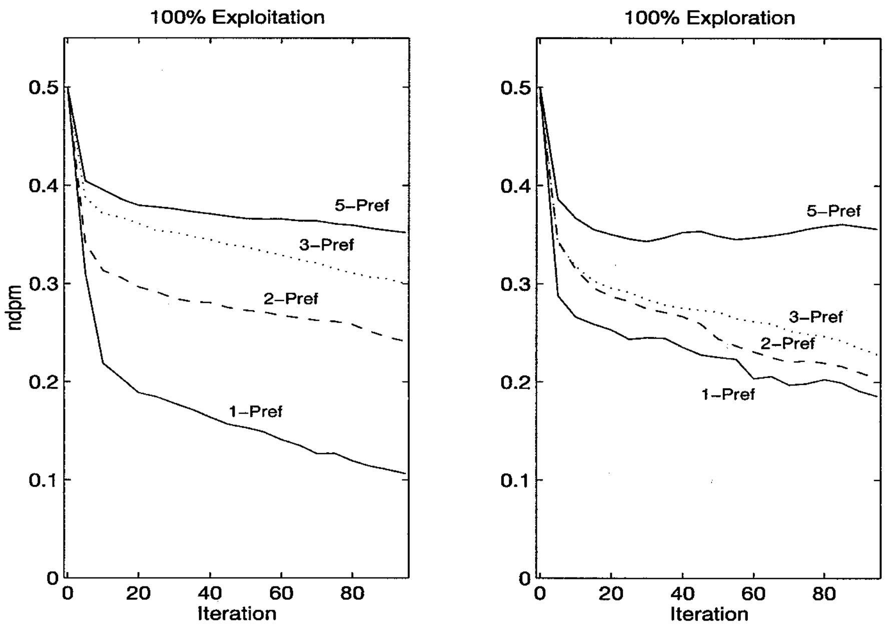

Figure 10. Learning curves for users with increasingly complex preferences.

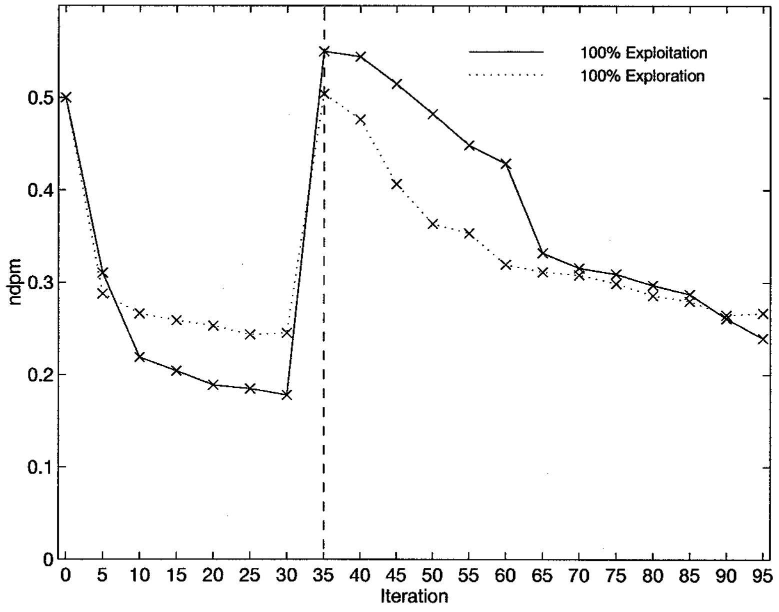

Figure 11. Exploitation vs. Exploration document selection strategies for 1-pref users where preferences change before iteration 35. Graph shows *ndpm* values averaged over all 10 possible users.

In general as the number of preferences is increased, learning becomes more difficult, and the simple linear techniques employed become less viable. Figure 10 shows how learning is less successful with more complex preference structures.

4.5. CHANGING USER PROFILES

So far, only static user preferences have been considered. However, in a realistic setting for a recommender system users' interests will be in constant flux. Many different simulations of user preference changes can be imagined, even within the limitations of the n-pref user preference structures defined earlier. In the following experiment, a very simple, drastic change is modeled, which provides a significant challenge to the learning system. A topic from the least preferred equivalence class is promoted to be the most preferred topic. Thus a topic the user previously found among the least interesting becomes the most interesting. An example of such a change in the preference structure for a 1-pref user is:

$$T_1 \succ \{T_2, \ldots, T_{m-1}, T_m\} \longrightarrow T_m \succ \{T_1, T_2, \ldots, T_{m-1}\}. \qquad (13)$$

Since there is no interaction between users, all of their preferences are changed synchronously, so that the effects will be maximally observable. Figure 11 compares learning curves for the pure exploitation and pure exploration strategies, for 1-pref users (compare to Figure 5, which is the same experiment but without the preference changes). It can be seen that the 'lag' is much greater for the exploitative strategy, since it is unlikely to deliver documents from the newly preferred topic.

Normally, user interest changes are handled by information filtering systems with some kind of function lessening the influence of older preference judgments, e.g., (Allan, 1996). In other words, there would usually be a change to the learning algorithm. In contrast, we show here that a change to the document selection strategy can also increase responsiveness to user changes, without necessitating assumptions about the longevity of user interests. It may be more efficient still to alter the document selection strategy when there is evidence that user interest changes are occurring, for instance if there is an increase in the disparity between predicted and actual rankings. Alternatively, deploying an exploration/exploitation slider (as described earlier) allows users to manually broaden the selection of documents when their interests change. Thus, another interpretation of the experiment above is that exposing such a control to the user is an appropriate measure to improve responsiveness to interest changes.

5. Related Work

At first glance the text recommendation task may seem similar to the 'ad hoc' retrieval, filtering or routing settings, commonly studied in the IR community, so we shall briefly explicate the differences. To ground the discussion, the particular definitions of these three tasks used for the TREC conferences are assumed (specifically TREC-5, the latest at time of writing, Harman, 1996).

The ad hoc retrieval task requires the user to formulate a query for an IR system which accesses a particular corpus of documents. In response, the system ranks the documents according to relevance to the query. The query represents a short-term information need, which will hopefully be satisfied by the top-ranked documents. In contrast, the recommendation task does not require a query to be formulated. Furthermore, discrete, unordered sets of documents are delivered, rather than a ranking over an entire corpus. Finally, the iterative nature of the task is better suited to long-term, ongoing information needs.

More similar to the recommendation task are the routing and filtering tasks, where users are assumed to have long-term information needs (Belkin and Croft, 1992). In the routing task, the system is not provided with a query, but rather a set of sample documents which are relevant to a user's information need. Using these documents as training data, the system can then rank the remaining documents in the corpus. For the filtering task, the system is further required to classify the remaining documents into relevant or irrelevant categories. Variations of this task are often studied from a machine learning perspective, as instances of supervised classification problems (Yang, 1997). In contrast, a recommender system must select its own samples of training data to present to the user for feedback. Furthermore, the fact that the process proceeds in an incremental rather than batch fashion means that many of the techniques used in routing or filtering which rely on batch processing of the training data are no longer as appropriate. Since systems performing the filtering and routing tasks are not required to select their own training data, they are not exposed to the exploration/exploitation tradeoff. In this respect, the difference between these batch tasks and the recommendation task is analogous to that between supervised classification and reinforcement learning algorithms.

While appealing to the narrow definition of information filtering used for the TREC conferences clarifies some differences, in actuality the breadth of research in this field is considerable (Oard, 1997), with various proposed systems tackling the issues addressed in this paper from different perspectives. At the intersection of machine learning and IR lies the related problem of text categorization. Lewis et al. have applied active learning to this problem, and in particular have shown how actively selecting training data for classifiers can improve performance considerably (Lewis and Gale, 1994; Lewis and Catlett, 1994). Their technique of *uncertainty sampling* resembles our exploration strategy. Gradient descent schemes similar to the one we describe have been successfully applied to text categorization (Lewis et al., 1996; Ng et al., 1997). Sumner and Shaw (1996) have shown that the adaptive linear model compares favorably to a probabilistic relevance feedback method on a two-stage retrieval task. The problem of learning orderings rather than classifications has been studied, with an application to Web retrieval (Cohen, et al., 1997). From an empirical IR standpoint, Allan (1996) has experimented with incremental feedback, where user judgments are available a few at a time. Bookstein (1983) has presented decision theoretic models which explicitly consider the utility of sets of documents retrieved by an IR system, rather than evaluating each document separately. A common application of user modeling to information filtering is to learn user models for filtering of articles from some constant stream, for example electronic newsgroups or internal company memos. Approaches include incremental construction of neural networks (Jennings and Higuchi, 1993), using latent semantic indexing to define dimensions for user profiles (Foltz and Dumais, 1992), genetic algorithms (Sheth and Maes, 1993) and using the minimum description length principle to find good models (Lang, 1995). A similar application is the discovery and recommendation of Web pages, which has motivated multiagent systems to represent the interests of single users (Moukas and Zacharia, 1997) or populations of users (Balabanović, 1997).

The parameterized *maximal marginal relevance* (MMR) measure (Frederking et al., 1997) as been proposed as a way of codifying the tradeoff between topical relevance and novelty, where in this case novelty is defined with respect to the other documents in the recommended set. Varying the parameter representing this tradeoff has resulted, in small-scale tests, in similar performance to the exploration/exploitation mixes reported in this paper, despite the fact the MMR measure takes no account of documents seen *before* the current set of recommendations, whereas the pure exploration strategy takes no account of other documents *within* the current set of recommendations. In fact the original motivation for the MMR measure was as a way to organize large search results from retrieval operations, rather than as a document selection strategy for a recommender system.

The systems described so far learn mainly statistical or instance-based representations, which model what has been called the 'shallow semantics' (Jennings and Higuchi, 1993) of the users' interests by using only words from the documents. In contrast, more frame-like user models refined from existing stereotypes have been proposed (Brajnik et al., 1987), and combined with semantic networks that represent relationships among words found in documents (Minio and Tasso, 1996; Asnicar and Tasso, 1997). These systems attempt to also infer higher-level knowledge of users and documents, often by attempting to assign values to some predetermined set of attributes.

Collaborative recommender systems form a related strand of research. A collaborative system will recommend an item based on the judgments of other users, rather than on the content of the item itself. Typically, the user model will be simply an unprocessed record of past preferences that can be matched with the user models of other users of the system. Common domains for such systems are movies (Hill et al., 1995; Fisk, 1996) or music (Shardanand and Maes, 1995). Systems also exist which tackle the recommendation of documents, e.g., (Resnick et al., 1994), and for that domain the incorporation of content analysis techniques (as commonly used for information filtering) has been suggested in order to alleviate some of the inherent difficulties (Balabanović and Shoham, 1997). In general, collaborative techniques utilize the variability of the user population in place of explicit exploratory strategies.

6. Summary

The simulation environment developed in this paper takes a step beyond empirical studies that consider binary classification schemes. We model users as having multilevel preferences among editorial categories, clusters of documents judged by human editors to be topically relevant to the same topic. In order to deliver recommendations to such users, a gradient descent learning algorithm that can make use of relative rather than absolute feedback is employed. The remaining design decision when creating a recommender system is the choice of document selection strategy, and this has been the focus of our experiments.

For users with simple single-topic preferences, an exploitative strategy has turned out to be best; for more complex users such as those with triple-topic preferences, there is a tradeoff between faster learning of user models and delivery of more preferred documents. As increasingly complex user preferences are simulated, the learning rate worsens.

In our future work we hope to investigate the suitability of recommender systems where users have control of their own exploration/exploitation settings. A practical implementation for such a control is developed in this paper, as a combination of a standard IR exploitative document selection strategy and a novel exploratory strategy.

Acknowledgments

This research was supported by NSF grant IRI-9503109. We are grateful to Geoff Webb, Alfred Kobsa and several anonymous reviewers for valuable comments.

References

Allan, J.: 1996, Incremental relevance feedback for information filtering. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, 270–278.

Allen, R. B.: 1990, User models: theory, method and practice. *International Journal of Man-Machine Studies* **32**, 511–543.

Asnicar, F. A. and Tasso, C.: 1997, ifWeb: a prototype of a user model-based intelligent agent for filtering and navigation in the World-Wide Web. In *UM97 Workshop on Adaptive Systems and User Modeling on the World Wide Web, 6th International Conference on User Modeling*. Sardinia, Italy.

Balabanović, M.: 1997, An adaptive web page recommendation service. In *Proceedings of the 1st International Conference on Autonomous Agents*. Marina del Rey, CA, 378–385.

Balabanović, M. and Shoham, Y.: 1997, Fab: Content-based, collaborative recommendation. *Communications of the ACM* **40**(3), 66–72.

Belkin, N. J. and Croft, W. B.: 1992, Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM* **35**(12), 29–38.

Bollman, P. and Wong, S.: 1987, Adaptive linear information retrieval models. In *Proceedings of the 10th International ACM/SIGIR Conference on Research and Development in Information Retrieval*, New Orleans, LA, 157–163.

Bookstein, A.: 1983, Information retrieval: A sequential learning process. *Journal of the American Society for Information Science* **34**(5), 331–342.

Brajnik, G., Guida, G. and Tasso, C.: 1987, User modeling in intelligent information retrieval. *Information Processing & Management* **23**(4), 305–320.

Carbonell, J.: 1983, The role of user modeling in natural language interface design. *Technical Report CMU-CS-83-115*, Carnegie-Mellon University, Department of Computer Science.

Cohen, W. W., Schapire, R. E. and Singer, Y.: 1997, Learning to order things. In *Advances in Neural Information Processing Systems 10*.

Fisk, D.: 1996, An application of social filtering to movie recommendation. *BT Technology* **14**(4), 124–132.

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.

Frederking, R., Mitamura, T., Nyberg, E. and Carbonell, J.: 1997, Translingual information access. In *Proceedings of the AAAI Spring Symposium on Cross-Language Text and Speech Retrieval*. Stanford, CA.

Fuhr, N.: 1989, Optimum polynomial retrieval functions based on the probability ranking principle. *ACM Transactions on Information Systems* **7**(3), 183–204.

Graber, D. A.: 1988, *Processing the News: How people tame the information tide*. Longman Inc.

Harman, D.: 1996, Overview of the fifth text retrieval conference (TREC-5). In *Proceedings of the 5th Text REtrieval Conference*, Gaithersburg, MD.

Hill, W., Stead, L., Rosenstein, M. and Furnas, G.: 1995, Recommending and evaluating choices in a virtual community of use. In *Conference on Human Factors in Computing Systems (CHI'95)*. Denver, CO.

Jennings, A. and Higuchi, H.: 1993, A user model neural network for a personal news service. *User Modeling and User-Adapted Interaction* **3**, 1–25.

Koller, D. and Sahami, M.: 1996, Toward optimal feature selection. In *Proceedings of the 13th International Conference on Machine Learning*, Bari, Italy.

Koster, M.: 1994, A Standard for Robot Exclusion. `http://info.webcrawler.com/mak/projects/robots/robots.html`.

Lang, K.: 1995, Newsweeder: learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, Tahoe City, CA.

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.

Lewis, D. and Catlett, J.: 1994, Heterogenous uncertainty sampling for supervised learning. In *Proceedings of the 11th International Conference on Machine Learning*, 148–156.

Lewis, D. and Gale, W.: 1994, A sequential algorithm for training text classifiers. In *Proceedings of the 17th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 3–12.

Lewis, D. D., Schapire, R. E., Callan, J. P. and Papka, R.: 1996, Training algorithms for linear text classifiers. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, 298–306.

Minio, M. and Tasso, C.: 1996, User modeling for information filtering on internet services: exploiting an extended version of the UMT shell. In *UM96 Workshop on User Modeling for Information Filtering on the World-Wide Web, 5th International Conference on User Modeling*. Kailua-Kona, HI.

Moukas, A. and Zacharia, G.: 1997, Evolving a multiagent filtering solution in amalthaea. In *Proceedings of the 1st International Conference on Autonomous Agents*. Marina del Rey, CA, 394–403.

Ng, H. T., Goh, W. B. and Low, K. L.: 1997, Feature selection, perceptron learning, and usability case study for text categorization. In *Proceedings of the 20th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 67–73.

Oard, D. W.: 1997, The state of the art in text filtering. *User Modeling and User-Adapted Interaction* **7**(3), 141–178.

Pazzani, M., Muramatsu, J. and Billsus, D.: 1996, Syskill & Webert: Identifying interesting web sites. In *Proceedings of the 13th National Conference on Artificial Intelligence*, Portland, OR.

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*, Chapel Hill, NC.

Resnick, P. and Varian, H. R.: 1997, Recommender systems: introduction. *Communications of the ACM* **40**(3), 56–58.

Rich, E.: 1979, User modeling via stereotypes. *Cognitive Science* **3**, 329–354.

Roberts, F.: 1979, *Measurement Theory*. Addison-Wesley.

Robertson, S. and Sparck Jones, K.: 1976, Relevance weighting of search terms. *Journal of the American Society of Information Science* **27**, 129–146.

Robertson, S. E.: 1977, The probability ranking principle in IR. *Journal of Documentation* **33**(4), 294–304.

Rocchio, Jr., J.: 1971, Relevance feedback in information retrieval. In G. Salton (Ed.): *The Smart System–Experiments in Automatic Document Processing*. Prentice Hall Inc., 313–323.

Rorvig, M. E.: 1988, Psychometric measurement and information retrieval. In M. E. Williams (Ed.): *Annual Review of Information Science and Technology*, Vol. 23. Elsevier Science Publishers B.V., 157–189.

Rosenblatt, F.: 1960, On the convergence of reinforcements in simple perceptrons. Technical Report VG-1196-G-4, Cornell Aeronautical Laboratory, Ithaca NY.

Rucker, J. and Polanco, M. J.: 1997, Siteseer: Personalized navigation for the web. *Communications of the ACM* **40**(3), 73–75.

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.

Schamber, L. and J. Bateman: 1996, User criteria in relevance evaluation: Toward development of a measurement scale. In *Proceedings of the Annual Meeting of the American Society of Information Science*. Baltimore, MD, 218–225.

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)*. Denver, CO.

Sheth, B. and Maes, P.: 1993, Evolving agents for personalized information filtering. In *Proceedings of the 9th IEEE Conference on Artificial Intelligence for Applications*. Orlando, FL.

Stadnyk, I. and Kass, R.: 1991, Modeling decision making of usenet news readers. *Technical Report CFAR-91-003*. EDS Center for Advanced Research.

Sumner, Jr., R. G. and Shaw, Jr., W.: 1996, An Investigation of relevance feedback using adaptive linear and probabilistic models. In *Proceedings of the 5th Text Retrieval Conference*. Gaithersburg, MD.

Taylor, R. S.: 1962, The process of asking questions. *American Documentation* **13**(4), 391–396.

Wang, P. and Soergel, D.: 1996, Beyond topical relevance: Document selection behavior of real users of IR systems. In *Proceedings of the 59th Annual Meeting of the American Society of Information Science*. Baltimore, MD, 87–92.

Wong, S. and Yao, Y.: 1990, Query Formulation in Linear Retrieval Models. *Journal of the American Society for Information Science* **41**(5), 334–341.

Wong, S., Yao, Y. and Bollman, P.: 1988, Linear structure in information retrieval. In *Proceedings of the 11th International ACM/SIGIR Conference on Research and Development in Information Retrieval*, Grenoble, France, 219–232.

Wong, S., Yao, Y., Salton, G. and Buckley, C.: 1991, Evaluation of an adaptive linear model. *Journal of the American Society for Information Science* **42**(10), 723–730.

Yang, Y.: 1994, Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 13–22.

Yang, Y.: 1997, An evaluation of statistical approaches to text categorization. *Technical Report CMU-CS-97-127*. Carnegie Mellon University, School of Computer Science.

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.

Author’s Vita

Marko Balabanović
Stanford University, Dept. of Computer Science, Gates Building 1A, Stanford CA 94305-9010, USA.

Marko Balabanović is a Ph.D. candidate in Computer Science at Stanford University. He received his B.A. in Computer Science from the University of Cambridge in 1990, and his M.S. degree in the same field from Stanford University in 1996.

His primary interests lie in the areas of adaptive information retrieval and human-computer interaction, although previous research has included work on mobile robot architectures, at Stanford University, and applications for mobile computers, at the Xerox Research Center, Europe. His paper reports on part of his thesis research on text recommender systems.