Filtering Medical Documents Using Automated and Human Classification Methods

J. Mostafa*
School of Library and Information Science, Indiana University, Bloomington, IN 47405-1801. E-mail: jm@indiana.edu; and Computer and Information Science, Purdue University School of Science at Indianapolis, 723 W. Michigan Street, SL280, Indianapolis, IN 46202

L. M. Quiroga
School of Library and Information Science, Indiana University, Bloomington, IN 47405-1801. E-mail: lquiroga@indiana.edu

M. Palakal
Computer and Information Science, Purdue University School of Science at Indianapolis, 723 W. Michigan Street, SL280, Indianapolis, IN 46202. E-mail: mpalakal@indyvax.iupui.edu

The goal of this research is to clarify the role of document classification in information filtering. An important function of classification, in managing computational complexity, is described and illustrated in the context of an existing filtering system. A parameter called classification homogeneity is presented for analyzing unsupervised automated classification by employing human classification as a control. Two significant components of the automated classification approach, vocabulary discovery and classification scheme generation, are described in detail. Results of classification performance revealed considerable variability in the homogeneity of automatically produced classes. Based on the classification performance, different types of interest profiles were created. Subsequently, these profiles were used to perform filtering sessions. The filtering results showed that with increasing homogeneity, filtering performance improves, and, conversely, with decreasing homogeneity, filtering performance degrades.

Introduction

On the Internet, using Listservs, Usenet news, FTP, or WWW tools, documents can be easily published and made available to millions of users. This convenience in many ways has become a curse. Due to lack of strict constraints on the number and type of documents that can be published, the Internet document universe is highly dynamic—it changes continuously. From the user's perspective, this makes the task of finding or selecting useful documents extremely difficult. To give the user more control over dynamic document sources, such as those typically found on the Internet, commercial organizations and research groups recently designed and implemented various information filtering (IF) systems (Konstan et al., 1997; Maes, 1994).

IF systems deal with the problem of prioritizing or minimizing dynamic document sets based on the long-term interests of users. They are similar to information retrieval (IR) systems in many respects. For example, they often use similar document representation and matching techniques (Belkin & Croft, 1992). However, they differ from IR systems in two important ways: (1) IF systems deal with long-term user interest represented as interest profiles, whereas IR systems usually deal with short-term interest represented as queries; and (2) IF systems deal with continuous streams of documents with varying content, whereas IR systems operate on relatively static document collections. These differences related to interest profiles and dynamic document sources require IF systems to manage complexities that IR systems generally do not deal with.

In this article, we focus on IF systems, with the particular aim of clarifying the role of classification. We begin in the next section by describing certain relevant concepts associated with filtering, and explaining how classification can aid in reducing computational complexity. We review recent research in the subsequent section, and define the research problem. Following this, we present an overview of an operational filtering system that was applied to investigate the problem empirically. Two algorithms for automating the classification process are then described. Next, the research methodology is presented,

* To whom all correspondence should be addressed.

Received May 5, 1997; revised September 30, 1997; accepted February 4, 1998.

with explanation of the relevant parameters that were used for independent and dependent variables. This is followed with a section on research results and analyses. We conclude the article with a summary of the primary findings and description of future research directions.

Filtering Process and Concepts

Document classification, categorization, and clustering are significant concepts in the way we framed the filtering process. Unfortunately, in the filtering literature, these terms are sometimes ambiguously presented. Hence, first we operationalize these terms and then describe their specific roles in the filtering process. We define classification as a process that produces mutually exclusive groups of documents. That is, once membership of a document in a particular class is established, the document can only belong to that class and to no other class. In contrast, we define categorization as a process that permits membership of a document in multiple classes (Jacob, 1991). To conduct classification, we apply a list of classes called a scheme. Humans have produced many such schemes covering diverse topical areas. One way to produce the scheme automatically, involves agglomerating related documents in a large set, identifying the most "representative" or "central" document in each subset, and using the representative documents as classes to generate the scheme. We refer to the process of identifying prospective classes as clustering.

To reduce the overall complexity of the filtering process, we propose multi-level decomposition (described below). In this process, a document-grouping step based on a fixed number of groups is introduced as an intermediate level. The rationale is that the number of document groups is generally going to be fewer than the number of incoming documents, and dealing with a fixed number of document groups instead of a dynamic document stream would reduce complexity. For grouping documents, a categorization process can be applied, however, this would imply that documents can belong to multiple groups. To further simplify the process, we apply, instead, a classification process whereby each document can belong to a single group only. In the actual classification step (i.e., assignment of classes to documents), a classification scheme is used. In this study, we aim to compare the utility of a human classification scheme with an automatically produced scheme in executing this intermediate step. To deal with the medical area, we selected a list of headings from the Medical Subject Headings (MeSH) and treated the list as the human classification scheme.[1] To automatically generate a classification scheme, a corpus of documents called a training set was used. First, distinctive terms were extracted from the corpus using an

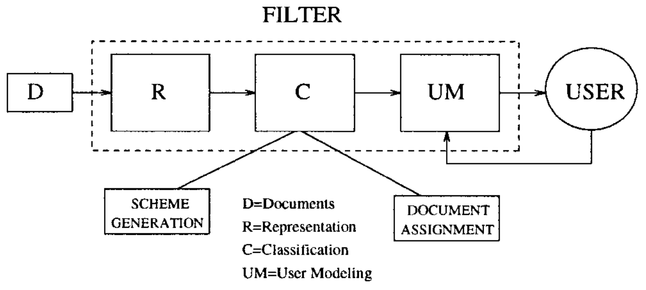

FIG. 1. Model of the filtering process.

automated process. Collectively these terms, referred to as a thesaurus, were then used to convert the documents to more compact representations. Each document representation is an array or a vector containing the frequency information related to distinctive term/s found in the document. Subsequently, a clustering algorithm was applied on the training document vectors. The function of the clustering algorithm was twofold: (1) It grouped document vectors using similarity measures that considered frequency of terms present in the documents, and (2) for each group, it identified a "central" vector formally known as the cluster centroid (Salton & McGill, 1983). The primary outcome of this step are the centroids, as these can be applied to group new documents from another source or stream. The complete set of cluster centroids, thus identified, was used as the automatically produced classification scheme.

Multi-Level Filtering

Filtering involves mapping documents to their respective relevance values. Formally, it can be posed as a map $f : \mathcal{D} \rightarrow \mathcal{R}$, where $\mathcal{D}$ is the document set, $\mathcal{R}$ represents relevance assessment of users, and $f(d)$ corresponds to the relevance of a document d. Given that such a map is known, any document in $\mathcal{D}$ can be rank ordered or pruned using the relevance assessment of the user. In IF systems, f however is not known a priori and must be established with user-interaction. For a large and continuously changing $\mathcal{D}$, establishing f directly based on individual documents is an extremely computationally demanding task. If, however, f is decomposed into two functions such that the first function is a map $f_1 : \mathcal{D} \rightarrow \{C_1, \ldots, C_m\}$ and the second function is a map: $f_2 : \{C_1, \ldots, C_m\} \rightarrow \mathcal{R}$, then the overall computational complexity can be considerably reduced. This is because the set of classes $\{C_1, \ldots, C_m\}$ is generally much smaller than the incoming document set and document relevance is assessed indirectly in terms of classes.

Recent Research and Research Problem

Based on a multi-level decomposition of the filtering process into the functions f_1 and f_2, we developed a model of filtering (see Fig. 1) and implemented a system called Smart Information Filtering Technology for Electronic Re-

[1] MeSH is a controlled vocabulary list created and maintained by the National Library of Medicine. MeSH is a categorization aid and, usually, multiple MeSH headings are assigned to a single document. However, in this study, only a single MeSH heading was maintained with each document, and the list of these headings was treated as a classification scheme.

sources (SIFTER). Using SIFTER, we performed empirical analysis of filtering in the area of computer and information science, and found that our system performs very well under realistic conditions (Mostafa, Mukhopadhyay, Lam, & Palakal, 1997). In the course of conducting our work, we observed that automated classification plays a crucial role in filtering, but its contribution to the filtering process is not well understood. Hence, we became interested in analyzing classification performance and in relating this performance to filtering in meaningful and useful ways.

Previously, we reviewed work on filtering in Mukhopadhyay et al. (1996) and Lam, Mukhopadhyay, Mostafa, and Palakal (1996). A more extensive and recent survey of filtering research can be found in Mostafa et al. (1997). A thorough comparison of filtering to IR research is offered by Oard and Marchionini (1997). Additionally, ongoing coverage of this area from both technological and scholarly perspectives can be tracked by investigating the excellent WWW site maintained by Oard (1997). Instead of duplicating these various sources, we begin with a brief summary of the major threads in filtering research. Later, we compare filtering with a particular branch of IR, and review literature in the area of automated document classification.

A key function in filtering is the acquisition of interest profiles. A line of work proposes knowledge engineering to create "user stereotypes" to generate the initial profiles. Rich's (1983) work on the Grundy system is one of the pioneering efforts. Approaches proposed by Brajnik, Guida, and Tasso (1990) and Shapira, Shoval and Hanani (1997) are more recent examples of application of stereotypes. A problem with stereotypes is that intensive knowledge engineering is necessary, usually on the part of the system administrator. Another branch of the research uses supervised learning techniques to acquire the profile in a batch mode. For example, the NewsWeeder system (Lang, 1995) relies on document ratings provided by the users as training examples. This system nightly applies a machine learning algorithm to generate updated profiles for the next day. Another type of filtering does not depend on extensive prior information, instead it relies on ongoing and explicit user involvement. The InfoScope (Fischer & Stevens, 1991) is such a system and it uses heuristic rules associating common patterns of usage (e.g., number of sessions, newsgroups read, frequencies of relevant terms, etc.) to appropriate actions. To refine profiles in InfoScope, users must add or remove terms from the profile and they must set appropriate rule-triggering thresholds. The requirement of explicit user involvement in rule-management is somewhat demanding and such rule-based approaches may be too "brittle" to support efficient profile adaptation. A similar filtering theme that depends on the involvement of many users in tasks such as document rating, voting, or selecting is generally referred to as collaborative filtering. A significant collection of research efforts along this line can be found in Arnheim (1996) or in the recent special issue of *Communications of the ACM* (editor: Diane Crawford, 1997).

An allied and an important issue in filtering is how the information to be filtered is actually acquired. In our approach, we assumed the user owns an existing document source that continuously receives new documents (e.g., electronic mailbox). In many other filtering environments, a centrally shared repository is assumed. For example, numerous filtering systems have been created for Usenet news. The centralization of the data can promote better control and predictability. However, chances of acquiring relevant documents can be improved if multiple sites or resources are used. In another approach, the filtering system "seeks" out multiple distributed resources to acquire related information. The Softbot system described by Etzioni and Weld (1994) is such a system. When given a high-level information need specification, it can select prospective Internet locations, execute searches, and collate the information that satisfies the specification. Better filtering service may also be attained by increasing the autonomy of the filtering system—delegating more responsibility to it and designing it to function in a more "pro-active" mode. Pattie Maes (1994) described intelligent agents that conduct filtering to manage meeting schedules, E-mails, and news. Collaboration among multiple filtering agents is a more recent focus of research. For example, among the six major national digital library initiatives, the University of Michigan group is explicitly involved in studying the coordination of multiple specialized agents working together to collect filtered information for users (Atkins et al., 1996). The underlying framework for collaboration is market-based resource allocation where information services are modeled as economic activities to optimize user services and minimize computational demands. Presently, our approach to filtering relies on a single system ("agent") dedicated to a single user. However, in the near future, we aim to investigate the utility of multi-agent collaboration in distributed environments.

With respect to IR research, our work on filtering has certain similarities to the TREC (Text Retrieval Conference) initiatives that deal with "routing tasks." Given a set of topics (information need descriptions) and known relevant and non-relevant documents for those topics, the routing task involves generating query representations or profiles that can be used to predict the relevance of new documents for a particular topic. A variety of techniques have been proposed and implemented for generating routing queries, ranging from Rocchio relevance feedback (Allan, 1996) to connectionist learning approaches (Boughanem & Soule-Dupuy, 1997). The overall performance in routing tasks in the recent TREC-5 experiments shows average precision at retrieval cutoff of 30 documents to reach as high as 53% (Beaulieu et al., 1997).

The TREC routing experiments, taken as a whole, offer useful insights into key challenges associated with profile acquisition. Some of the basic techniques developed for routing can certainly be applied in filtering. Our present research focus, however, is different from TREC routing experiments in certain ways. In TREC routing, it is assumed that a set of training documents with relevance judgments is available a priori. Whereas in SIFTER, although a training document set is used, we do not assume that relevance judgments are available a priori for the documents. In our previous work, we developed a user-modeling technique for profile acquisition that does not rely on a priori judgments but collects relevance judgments online based on the incoming document stream. We demonstrated that SIFTER can acquire different types of profiles quickly and with little computational overhead (Mukhopadhyay et al., 1996). One particular way that we tackled the profile learning problem was to base the profile on an intermediate set of classes instead of directly on documents (the profile acquisition task was decomposed into the functions f_1 and f_2). While in TREC routing,[2] direct acquisition of profiles from the documents generally receive the primary emphasis, in our present work, we assume an explicit intermediate classification level, and analysis of its influence on filtering is given the primary focus.

A number of recent studies have attempted automated document classification. Cheng and Wu (1995) described the development of the Automated Classification System (ACS) that can conduct classification at 80% accuracy level. The ACS system did not attempt to generate a new classification scheme, rather it utilized term vector representation of classes derived from an existing scheme (Dewey Decimal Classification). It focused on classification of only books, based on the title and chapter headings. Larson (1992) described an automated classification approach that also relied upon an existing classification scheme (Library of Congress Classification). In this research, each class was represented as a vector containing frequencies of tokens (class numbers and term stems derived from subject headings and titles) appearing in documents that had been previously assigned to that class. Individual documents were also represented with vectors, containing frequencies of tokens found in their MARC records. This research presented experimental data on several variants of weighting and matching techniques as ways to classify books. It found that items represented using LC subject headings alone with probabilistic matching produced the best classification results (approximately 47% accuracy). Yang and Chute (1994) described an approach in which document membership in multiple classes (categorization) can be determined by using an automatically produced classifier. The classifier was created using the least-squares-fit method, in which the mapping equation was derived based on a training set of document vectors (term weights) and vectors of previously assigned categories to documents (binary values representing presence or absence of categories). Yang and Chute (1994) showed that their method was superior in matching documents to target classes when compared to purely string-based matching and the standard *tf.idf*-based (term frequency multiplied with inverse document frequency) matching. In their research, however, no new classes were generated or used. Further, the training method was supervisory in nature, requiring initial availability of target classes for the system to learn to classify. May (1997) performed filtering of E-mail messages using four classes: Question, response, announcement, and administrative. His work concentrated on non-topical but otherwise relevant features of messages. The association between messages and classes relied on string-matching and it achieved approximately 46% accuracy. Finally, Jacobs and Rau (1990) described the design of a filtering system called SCISOR that operates on a news database (Dow Jones). This system incorporates both an explicit document classification component and a user interest component (queries) in the same system. This system was successful in attaining 90% recall and precision in identifying relevant documents out of a test set of 729 documents. SCISOR relies on natural language processing algorithms and preestablished knowledge-bases to perform filtering, and the results presented covered the domain of "acquisition and merger" in businesses.

Although the past research offered useful insights into document and class representation, it did not present sufficient evidence to clarify the role of automated classification in filtering. Additionally, in our work, we are interested in analyzing unsupervised classification. Such an approach implies that no prior classes are manually coded into the system, rather classes are discovered or learned without human intervention. The past work reviewed here did not directly address the role of the unsupervised classification approach in filtering.

The SIFTER system provides a platform to analyze classification and filtering as complementary processes of a single system. The modularity of the SIFTER system allows for convenient substitution of the classification scheme and comparative analysis of different classification approaches. In our view, such comparative analysis, especially between a human approach and an automated approach, can lead to better understanding of the relationship between classification and filtering in a number of ways. At a general level, by assuming that a human classification approach is most accurate (treated as control or baseline), we can investigate if an automated classification approach can approximate this performance. More specifically, it would be useful to know: How does improvement or degradation in classification influence filtering performance? Or, exactly how much accuracy is necessary in classification to ensure satisfactory filtering performance?

Fortunately, in established disciplines, such as medicine, large sets of documents exist that are already classified by human experts. This provided us an opportunity to perform new experiments to test the robustness of SIFTER (to filter documents in a different domain) and to also address some of the questions raised above.

[2] It should be noted that the most recent TREC-6 invited participation in a specific track on filtering, and in TREC-7, the routing tasks will be replaced by the filtering track.

Sifter System and Automated Classification

SIFTER is a fully implemented filtering system, with special features built-in for conducting evaluations. SIFTER uses the "message processing" convention for document filtering. That is, it assumes a predesignated document "bin" on the user's computer continuously receives documents from various sites on the network and, when invoked, its task is to present these documents to the user. In SIFTER, the major components of the filtering model, (1) Representation, (2) classification, and (3) user modeling and document presentation, shown in Figure 1, have been implemented. Here, we will present a brief overview of the system.

SIFTER Components

When executed, SIFTER checks the predesignated document bin for newly arrived documents. If new documents are found, it converts each document to a vector containing significant terms found in the document. We chose the vector-space model (Salton & McGill, 1983) for document representation because it has been widely tested. During this phase, an online thesaurus is used. In SIFTER, the thesaurus is an array T where each element contains a value-pair: An atomic token (a single word) and a numeric identifier. The main function supported by the thesaurus is efficient identification and utilization of domain-oriented vocabularies in the documents. The thesaurus, with some human control over the generation process, can also be used for synonymy control, such that if two tokens with equivalent semantic relationships are stored as elements e_1 and e_2, then they should be assigned the same numerical identifier, i.e., $T[e_1].id = T[e_2].id$. The system administrator can select the tokens from established control vocabulary sources such as indices or thesauri created by experts. In fact, in our previous work, for the area of computer and information science, we utilized the Association for Computing Machinery Computer Science Classification Scheme and the American Society for Information Science Thesaurus to create tokens. However, it is also possible to generate the tokens by applying automated thesaurus generation techniques (Salton & McGill, 1983). Regardless of the procedure used for token generation, they ultimately need to be entered into the system along with the appropriate numeric identifiers.

We apply the *tf.idf* approach (Equation 1) to establish the degree of importance of thesaurus-tokens found in individual documents in the incoming stream (Salton & McGill, 1983). To apply this technique, a table is generated offline containing total frequencies of all tokens in the thesaurus using a large collection of documents as a base. We used the 6,000 documents from the training set as a base (the training set is discussed later in the article). A separate and large base is especially useful in filtering because the incoming document stream may occasionally contain only a few documents. During the online filtering mode, another table is generated containing the frequencies of all unique tokens found in the newly arrived documents. Then, using the values in the two tables, the *tf.idf* formula is used to derive appropriate weights for terms in each document. The base file values are applied in the *idf* component of the formula, i.e., in $log(N/n_k)$, where N is the total number of documents in the base and the stream, and n_k is the number of documents in the base and the stream that contain the given token. In terms of the two-level functional decomposition, this representation phase produces a set of document vectors, $V_i s$, that make up the input space for the function f_1.

The classification module supports two distinct functions: Scheme generation and vector classification (we discuss this in detail later). Scheme generation is conducted as an offline process, based on a representative sample of document vectors $[S_1, \ldots, S_n]$, with the outcome being a set of clusters $[C_1, \ldots, C_k]$. Each cluster C_i is represented as a cluster centroid Z_i, hence, the scheme constitutes a set of such centroids. Semantically, the scheme can be viewed as a high-level grouping of concepts so that they form sub-areas or classes in the domain covered by the thesaurus. In SIFTER, each element in the vector Z_i represents a particular token identifier in the thesaurus, and the dimension of Z_i = number of unique token identifiers in the thesaurus. Thus, for example, the area of "Antibody Formation" would be represented, numerically, as a vector with high weights for the token identifiers associated with "Antibody" and "Formation" and all the other weights in the vector would be set to zero. This way of creating classes can be completely automated by applying a clustering algorithm on a representative document set and generating centroids for all the clusters found. In our previous work, we applied the *Maximin-Distance* clustering algorithm to conduct this operation (discussed below). However, assuming that established vocabularies and a set of classes are available for a domain, the vectors can also be manually produced and entered into the system. During the operational mode, the classification module classifies an incoming document vector as belonging to the class whose centroid has the minimum distance to the document vector. The measure used for computing the distance between each document vector V_i and class centroid Z_i is shown in Equation 2. The classification phase implements the map f_1. As its output, this phase generates class labels for all the documents that constitute the input space for the function f_2.

In the final phase, the user modeling module takes over. It treats the document clusters C_i as classes. Its main function is to determine the user's preference for the different classes C_i, and prioritize the incoming documents based on the classes, as well as the estimated user-preferences for the classes. To accomplish this, the system utilizes a profile learning algorithm adapted from the reinforcement learning area (Narendra & Thathachar, 1989). The learning algorithm, captures and updates an estimated relevance probability vector of preference values over the set of classes C_i. This vector, $\hat{d}$, is represented as an approximation of the idealized interest profile d. During the first invocation, $\hat{d}$ is initialized to contain only zeros because no preference in formation is available. After documents are presented, users are asked to rate each document on a binary scale of 0–1 (0 = not interested and 1 = interested). The relevance feedback provided for each document, is assumed to be a feedback value for the corresponding document class. Specifically, after several document presentation and relevance feedback cycles, $\hat{d}_i$ ($i = 1, \cdots, n$) would constitute the running average of the relevance values given by the user for documents belonging to class i. In addition to the $\hat{d}$ vector, the learning algorithm also uses an action probability vector $p = [p_i]$, such that p_i represents the probability that the class C_i is selected as the most relevant class. In the absence of any a priori knowledge, all elements of p vector are initially made equal to each other, i.e., p_i, ($i = 1, \ldots n$) are initialized to $1/n$. Both p and $\hat{d}$ vectors are updated during the learning process on the basis of relevance feedback. The p vector allows the learning scheme flexibility of exploring all the classes during the early period of use. After a certain number of iterations, the ranking (presentation order) of relevant documents would reflect the values of their corresponding classes in $\hat{d}$. If the user becomes satisfied with document-ranking, further relevance feedback would become unnecessary. However, relevance feedback can be continued and the internal profile, $\hat{d}$, can be modified in an ongoing fashion. This phase, completes the functional mapping of our model by implementing f_2.

External conditions may bring about changes in the user's interest, or a domain may change with the introduction of new topics. The former change can be characterized as "query drift," whereas the latter is "concept drift" (Allan, 1996). The filtering system must be able to adapt to these changes in a graceful fashion, without drastic or sudden degradation in service. Although, dealing with such changes is not the focus of this study, some comments on how SIFTER can handle such changes are offered here. In our previous work, as appropriate responses to these changes, we proposed certain tuning operations conducted to stabilize the SIFTER system (Lam et al., 1996; Mostafa et al., 1997). The query drift situation can occur at any time, with unanticipated or sudden change in the external environment. Because such a case can have an immediate effect on the filtering performance, it should be preferably handled online and automatically. We implemented a module that conducts Bayesian analysis on the relevance feedback data to detect shifts in the user's interest, and it can successfully adjust the profile to avoid filtering degradation. The concept drift condition, may be less frequent and its rate is generally slower. Presently, the SIFTER system does not use relevance feedback data to automatically modify the lower level components (i.e., the classes or the thesaurus), however, the system can be adjusted to deal with such changes in certain other ways. The classification scheme and the thesaurus can be regenerated periodically in a batch operation, using the last n documents as the new training set. Also, the modularity of the SIFTER system allows for convenient manual modification to the class structure or the thesaurus to reflect the changes in the domain.

The SIFTER system operates in the Sun Solaris UNIX and the Hewlett-Packard UNIX environments. SIFTER's filtering components were created using the C language. SIFTER's graphical user interface (GUI), with functions for document viewing, profile monitoring, and feedback collection, was created using the TCL/TK programming language. A logging facility in SIFTER can save data for each session, including list of document identifiers (before and after ranking), class labels and values in $\hat{d}$. In addition to an interactive mode, SIFTER also supports an autonomous filtering mode. In this mode, a user profile (d) can be directly entered into the system, and other parameters can be set (e.g., number and types of classes used or number of documents to be processed per session) to perform document filtering without user intervention. The profile d is used to establish document relevance and also to generate relevance feedback. For each class in d, a value between 0–1.0 (inclusive) is entered, signifying the level of interest for the class. The determination of document relevance and feedback is carried out probabilistically, so that documents belonging to classes with high interest values (near 1.0) would be more likely to be selected as relevant than documents belonging to classes with low interest values (near 0). In the autonomous mode, $\hat{d}$, the internally generated profile is still used as the basis for document ranking and presentation, because $\hat{d}$ is designed to be adaptive and, thus, it is more responsive to changes in filtering demands. More detailed description of the major components of SIFTER and results of usability evaluation can be found in Mostafa et al. (1997).

Automated Classification

Before new experiments could be conducted, we had to develop a method for automatic medical document classification. Particularly, we needed a vocabulary discovery procedure to identify tokens for the thesaurus. Further, based on the thesaurus, a new scheme had to be automatically generated. We describe two complementary techniques for performing these tasks.

Vocabulary Discovery

Below, we present an algorithm. Following that, we explain the underlying rationale and assumptions.

(1) Based on the complete training set of documents, generate a table. Each tuple in the table should contain information about each unique token per document. A tuple should contain a token, an identifier for the document containing the token, and the frequency of the token in the document.

(2) Calculate a weight for each token using the *tf.idf* formula:

$$W_{ik} = t_{ik} \times log(N/n_k) \quad (1)$$

where t_{ik} is the number of occurrences of token t_k in document i, $I_k = log(N/n_k)$ is the inverse document frequency of the token t_k in the training set, N is the total number of documents, and n_k is the number of documents in the training set that contain the given token t_k.

(3) To the table generated in (1), add a column containing a numerical ranking for each token, based on its weight per document. That is, the token with the highest weight in each document should receive the rank of 1, the second highest should receive the rank of 2, and so on.

(4) Sort the table produced in (3) by rank and token. Extract tokens from the table that appeared in at least D documents and were ranked between 1 and R (R should be a small number to ensure selection of highly ranked terms).

We have selected a relatively straightforward technique, the *tf.idf* approach, for token weighting. Various token weighting and refinement techniques are actually available in IR, ranging from the statistical procedures that calculate keyword distribution to more sophisticated techniques that rely on analysis based on natural language processing (NLP) algorithms. However, the general and somewhat surprising finding in IR is that the keyword distribution-based approaches are almost as effective as the more sophisticated approaches (Lewis, 1992).

Two basic assumptions were made regarding the training domain. First, documents from a given domain can be divided into subgroups based on their topical coverage. Second, not all tokens are equally representative of the topic/s covered in a subgroup. More specifically, in each document subgroup, a subset of the tokens is likely to have more resolving power or be more discriminatory (Salton & McGill, 1983) with relation to particular topics as compared to other tokens appearing in the subgroup. The latter assumption is similar to the assumption behind the *tf.idf* approach to term weighting based on a document collection (Salton & McGill, 1983). Here, instead of the whole collection, we apply it to document subgroups. The goal of the algorithm is to extract the token subsets that are relatively superior in representing relevant topics covered in the document subgroups. Relevant topics in the document subgroups are equated with classes, and the aim is to identify prospective tokens for the thesaurus that have strong semantic association with the classes. Although, this algorithm does not directly use classification information, such information can still be useful. The algorithm can be tweaked—its performance enhanced—if the scope or breadth of the individual documents subgroups present in the training set is known or can be estimated (specifically, for fixing an upper-bound for D). In our study, the training set documents contained only title and abstract. In creating the training set, however, we exercised control over the scope of document subgroups ($D = 400$) and limited the overall topical coverage to cell biology.

Scheme Generation

Highly ranked tokens produced from the above procedure (ranging between 1–10) were used to create a new thesaurus. Applying the new thesaurus, training set documents were converted to vectors V, with the dimension of V = number of unique token identifiers in the thesaurus, and each vector element representing a weight computed by using the Equation 1. Then, a heuristic unsupervised clustering algorithm, called *Maximin-Distance* algorithm (Tou & Gonzalez, 1974), was used to determine the centroids over this document vector space. In this algorithm, the centroids are generated in an iterative fashion. The distance between vectors is calculated using a formula based on cosine similarity (similar to Equation 2). Each element (a document) in the space is treated as a potential centroid. The first element is selected as the first centroid and saved. Next, the element farthest from the centroid is selected as the second centroid. Minimum distances of all the other elements to these two centroids are then calculated and saved. Among these, the element having the maximum distance is selected as a new centroid if it is an appreciable fraction of the distance between the two centroids. Minimum distances of the rest of the elements to the centroids are then determined and saved. Again, from these elements, the element having the maximum distance is selected. This particular element is chosen as a centroid if the distance is an appreciable fraction of the average of previous maximum distances. The last three steps are repeated, until no more centroids are found. A threshold value θ is used as the appreciable fraction, and this, in turn, controls the granularity or the number of clusters in the outcome. A high value for the threshold (near 1.0) would produce a smaller set of clusters than a low value. Generally, the outcome of this process is cluster centroids that are much fewer in number than the total number of documents in the original training set. In our research, the centroids represented the classes, and the complete set of classes is treated as the scheme.

After generating the scheme, it is entered into SIFTER as a set of vectors Z. During the online filtering operation mode, the classification module calculates for each document vector $V = [v_1, \ldots, v_t]$ its distance from the vectors $Z = [z_1, \ldots, z_t]$ using the formula below:

$$1 - \sum_{i=1}^{t} v_i z_i / \sqrt{(\sum_{i=1}^{t} v_i^2)(\sum_{i=1}^{t} z_i^2)}. \qquad (2)$$

This formula is based on the cosine similarity measure proposed in Salton and McGill (1983). The document is assigned to the class with the centroid producing the minimum distance, and the resulting class information is then passed on to the user profile learning module. Documents with exclusively zero weights in their corresponding V vectors (no token in the thesaurus matched with the document content) are assigned to a special class by the system called the "others" class.

Research Methodology

In terms of an empirical framework, the type of classification method (human or automated) was our independent variable, and filtering effectiveness was our dependent variable. In filtering, one of the major objectives is to identify those documents in the incoming stream that may be potentially relevant to the user. As described earlier, in the autonomous filtering mode, selection of relevant documents is directly based on the interest values specified in the profile d. In each session, among all the documents presented, SIFTER records the total number of relevant documents ranked between 1–10. This value, representing filtered documents per session (FDS), was one of the measures used for the dependent variable.

During the initial few sessions, the internally generated profile $\hat{d}$ has little or no information about the user's interest, hence, the ranking of relevant documents is generally poor. However, with time, SIFTER acquires more information, and ranking of relevant documents improves (placed consistently at, or near, the top). We used another parameter to measure our dependent variable, called normalized precision, to precisely track the overall ranking of all relevant documents identified in each session. Normalized precision and normalized recall are composite measures that are independent of the retrieval threshold used to distinguish the retrieved from the non-retrieved items, and they are applicable in systems that rank the retrieved documents (Salton & McGill, 1983). In SIFTER, all documents that are processed in a session are presented to the user in a ranked fashion, hence the separation of documents into retrieved and non-retrieved sets is artificial. In our experience with SIFTER, we found normalized precision and normalized recall to show similar trends, therefore, to reduce redundancy, we chose to use only the normalized precision measure:

$$Precision_{norm} = 1 - \frac{\sum_{i=1}^{REL} log\ RANK_i - \sum_{i=1}^{REL} log\ i}{log(N!/(N - REL)!REL!)} \quad (3)$$

In the above equation, N is total number of documents in the stream, REL represents total number of relevant documents, and $RANK_i$ is the ranking of the relevant document i in the final output (Salton & McGill, 1983).

Our independent variable has two levels: Human and automated. Here, we treated the human level as our control or baseline. That is, we assumed human developed classification is desirable and accurate. The automated level relies on a method of classification that is conducted independent of human intervention. To measure the performance of automated classification, we utilized the following formula:

$$Homogeneity_i = \frac{HMOD_i}{R_i} \quad (4)$$

In the above equation, i represents an automatically generated class. $HMOD_i$ is total number of documents from the modal human class classified into class i. R_i is the total number of documents classified into class i using the automated method.

For each automatically generated class, we considered the consistency with which documents from a particular human established class were classified into that class. Specifically, we considered the total number of documents classified into each automated class and their original human assigned class. Then, the most frequently appearing documents from a single human class was designated as the modal class. Subsequently, the ratio of the proportion of records in the modal class to that of the total records classified into the automated class was calculated. Finally, an overall homogeneity measure for the automated classification approach was produced by averaging the homogeneity scores of all automatically generated classes.

As user interaction was not the focus of this study, preestablished interest profiles were entered into the system as d and kept constant across the two levels of the independent variable. Values representing particular interests, i.e., the values to be specified for classes in d, were randomly established. To relate classification performance to filtering, profiles were created by selecting and grouping classes according to their homogeneity scores.

We utilized 7,500 document bibliographic records in our research. Topical coverage of documents was limited to 15 cell biology MeSH categories (Table 1). To facilitate comparison of classification performance, only a single MeSH category (henceforth referred to as a class) was used per document. Treating each class as a major descriptor label, these documents were downloaded from the Medline database. The document set was divided by randomly assigning documents into two subsets: (1) A 6,000-document training set (400 per class) for automatically discovering vocabularies and classes, and (2) a 1,500-document testing set (100 per class) for measuring classification and filtering performance. Documents in the training set only consisted of title and abstract—the goal was automatic classification scheme generation without relying on MeSH vocabularies. To conduct filtering, two versions of SIFTER were used. One version utilized the automatically generated scheme, while the other version used the 15-class MeSH cell biology

TABLE 1. MeSH categories.

CELL ADHESION
CELL COMMUNICATION
CELL DEATH
CELL MOVEMENT
CELL SURVIVAL
ENDOCYTOSIS
ANTIBODY FORMATION
AUTOIMMUNITY
IMMUNOCOMPROMISED HOST
CYTOTOXICITY IMMUNOLOGIC
IMMUNE TOLERANCE
IMMUNITY CELLULAR
REGENERATION
EVOLUTION
COMPLEMENT ACTIVATION

TABLE 2. Documents classified using MeSH.

Class	Total
CELL ADHESION	90
CELL COMMUNICATION	96
CELL DEATH	94
CELL MOVEMENT	98
CELL SURVIVAL	97
ENDOCYTOSIS	99
ANTIBODY FORMATION	86
AUTOIMMUNITY	96
IMMUNOCOMPROMISED HOST	97
CYTOTOXICITY IMMUNOLOGIC	96
IMMUNE TOLERANCE	99
IMMUNITY CELLULAR	99
REGENERATION	95
EVOLUTION	99
COMPLEMENT ACTIVATION	99
Total	1,440

scheme for classification. We modified the test set, used as input for the two different SIFTER versions, for compatibility with the type of classification desired. The input test set for the automated classification level only contained title, abstract, and a sequentially generated document number, while the input test set for the control level (MeSH) only contained a single class (one of 15) and a sequentially generated document number. Other than the basic difference in representation, the same number and the same document sequence were maintained in the two versions of the test set.

Research Results and Analysis

Below we discuss relevant findings related to all the stages of the research outlined above, including automatic generation of a scheme, classification of documents using the scheme, and filtering of documents based on both the new scheme and the established MeSH classes.

Automatically Produced Thesaurus and Scheme

Using the vocabulary discovery procedure on the training set, 725 tokens ranked between 1–10 were extracted. In this new set, we then summed individual document frequencies of tokens (number of documents each token appeared in) and sorted the set by the total document frequency per token. From this set, we removed a few ambiguous tokens (e.g., “med,” “tcr,” “cd,” etc.) whose lengths were generally <4 characters. This left us with a set where the highest ranked token was “complement” that appeared in 256 documents, and the lowest ranked token was “cord” that appeared in 16 documents. By using a value of 60 for D, we removed the rest of the tokens from the set and this left us with 50 tokens. We entered all these tokens into the thesaurus. Until this step, the generation of the controlled vocabulary required little or no human intervention. However, to gain further efficiency, we exercised some manual control in assigning unique identifiers to the thesaurus tokens. Among the 50 tokens in the thesaurus, eight tokens had other semantically equivalent tokens. Five of these were singular-plural cases and three were cases of variant forms. We grouped these tokens according to semantic equivalence and assigned them the same identifiers. Thus, in terms of unique identifiers, the thesaurus size was contracted to 42.

Next, the new thesaurus was applied along with the training set to generate classes based on the *Maximin-Distance* clustering. We set the threshold value θ to 0.999 as we wished to produce a small set of classes with broad scope. The resulting set contained 22 centroids. These centroids and a null vector (for the “others” class) were subsequently entered into the system as the automatically generated scheme.

To analyze the quality of the automatically produced scheme (first column in Table 3), we compared it with the 15 MeSH class set (Table 2). We found the new scheme compared well with the actual MeSH classes both at the lexical (token) level and at the semantic level. Among all the new classes, a subset of eight classes represented the original MeSH classes with high precision. These were: MIGRATION (MeSH = CELL MOVEMENT), APOPTOSIS (CELL DEATH), AUTOIMMUNE (AUTOIMMUNITY), RECEPTOR, ENDOCYTOSIS (ENDOCYTOSIS), REGENERATION (REGENERATION), EVOLUTION,

TABLE 3. Documents classified using automated classes.

Class	Total	Documents from modal MeSH class
REGENERATION	45	43
EVOLUTION, DNA	51	47
RECEPTOR, ENDOCYTOSIS	70	64
APOPTOSIS	81	69
TOLERANCE	39	33
PLASMA, MEMBRANE, COMPLEMENT, ACTIVATION	106	87
AUTOIMMUNE	86	64
INFECTED, IMMUNOCOMPROMISED	78	57
MIGRATION	51	36
SURVIVAL	37	26
MUSCLE	30	18
PRODUCTION, MOTILITY	29	17
DEATH, COMMUNICATION	42	24
CYTOTOXIC	75	41
SERUM, ANTIBODIES	64	30
TUMOR	27	12
TRANSPLANT	21	9
GROWTH, CELL, ANTIGEN, ADHESION	169	55
RESPONSE, IMMUNE, GENE, CLASS, CELL	144	41
CELL, BINDING	43	12
VIRUS	43	10
EXPRESSION	52	9
OTHERS (NULL)	57	12
Total	1,440	816

TABLE 4. Automatic classification homogeneity.

Automatic class	Homogeneity	Modal MeSH class
REGENERATION	0.96	REGENERATION
EVOLUTION, DNA	0.92	EVOLUTION
RECEPTOR, ENDOCYTOSIS	0.91	ENDOCYTOSIS
APOPTOSIS	0.85	CELL DEATH
TOLERANCE	0.85	IMMUNE TOLERANCE
PLASMA, MEMBRANE, COMPLEMENT, ACTIVATION	0.82	COMPLEMENT ACTIVATION
AUTOIMMUNE	0.74	AUTOIMMUNITY
INFECTED, IMMUNOCOMPROMISED	0.73	IMMUNOCOMPROMISED HOST
MIGRATION	0.71	CELL MOVEMENT
SURVIVAL	0.70	CELL SURVIVAL
MUSCLE	0.60	REGENERATION
PRODUCTION, MOTILITY	0.59	CELL MOVEMENT
DEATH, COMMUNICATION	0.57	CELL COMMUNICATION
CYTOTOXIC	0.55	CYTOTOXICITY IMMUNOLOGIC
SERUM, ANTIBODIES	0.47	ANTIBODY FORMATION
TUMOR	0.44	CYTOTOXICITY IMMUNOLOGIC
TRANSPLANT	0.43	IMMUNOCOMPROMISED HOST
GROWTH, CELL, ANTIGEN, ADHESION	0.32	CELL ADHESION
RESPONSE, IMMUNE, GENE, CLASS, CELL	0.28	IMMUNITY CELLULAR
CELL, BINDING	0.28	CELL ADHESION
VIRUS	0.23	IMMUNOCOMPROMISED HOST
EXPRESSION	0.17	CELL COMMUNICATION

DNA (EVOLUTION), SURVIVAL (CELL SURVIVAL), and INFECTED, IMMUNOCOMPROMISED (IMMUNOCOMPROMISED HOST). A subset containing six new classes also showed good semantic correspondence with the actual MeSH classes, but they were less precise. These included: CYTOTOXIC (MeSH = CYTOTOXIC IMMUNOLOGIC), TOLERANCE (IMMUNE TOLERANCE), PRODUCTION, MOTILITY (CELL MOVEMENT), MUSCLE (REGENERATION), SERUM, ANTIBODIES (ANTIBODY FORMATION), PLASMA, MEMBRANE, COMPLEMENT, ACTIVATION (COMPLEMENT ACTIVATION). The semantic relationship of the other eight new classes with the 15 MeSH classes appeared to be less direct or ambiguous. We also observed that some of the new and highly precise classes demonstrated an interesting relationship with the corresponding MeSH classes. These new classes contained tokens that were lexically different from the corresponding MeSH classes, yet their relationship to the MeSH classes were semantically strong. For example, the new classes APOPTOSIS, AUTOIMMUNE, and MIGRATION can be considered strongly related to the MeSH classes CELL DEATH, AUTOIMMUNITY, and CELL MOVEMENT, respectively.

Automated Classification

To analyze classification performance, our goal was to apply the new scheme to classify documents from the test document set. Before such analysis could be performed, however, we needed to refine and modify the test document set further. Among the documents in the test set, 31 duplicates were discovered and removed. We aimed to utilize the same documents for subsequent experiments to analyze filtering. Specifically, we wished to utilize the documents from the test set in a sufficiently large number of sessions that involved presentation and processing of new documents in each session. Fixing the number of new documents presented per session at 30, we determined that a maximum of 48 sessions could be executed requiring a total of 1,440 documents. Hence, the automated classification performance analysis was limited to only these 1,440 documents.

In Table 2, we show the classification results of 1,440 documents when SIFTER used the MeSH classes (control level). In Table 3, we present the classification results of the same documents based on the automatically generated scheme and the proportion of documents from the modal class. Overall, we found that the new scheme classified 1,383 (96%) documents into one of the 22 non-null classes. The average number of documents classified into the 22 new classes was 63. The overall homogeneity score (Equation 4) for all the new classes was 58%, and without the null class (“others”), it was 60%. When individual homogeneity scores of the new classes were considered, we found that the top six classes had higher than 80% homogeneity, the next eight classes had homogeneity scores between 50 and 75%, and the rest of the classes had less than 50% homogeneity scores (Table 4). Three hundred and ninety-two documents (27%) were classified using the best performing six classes, 428 documents (30%) were classified using the next eight classes, and, therefore, a combined total of 820 (57%) documents were classified at the level of 50% homogeneity or higher. The top 14 classes, in terms of homogeneity, also demonstrated strong or good semantic relationship with the corresponding modal MeSH classes (Table 4). The semantic relationship of these top performing classes with the modal MeSH classes closely mirrored the semantic relationship we found earlier, based on the direct scheme-to-scheme comparison.

Filtering Performance

The main goal in conducting the filtering experiments was to establish if differences in classification methods produce different filtering performance. A secondary goal was to establish if different profiles created using different subsets of the classes (automatically produced and human produced) lead to predictable variability in the filtering performance.

Six different profiles were created using various combinations of classes. For the automated classification level, two profiles were based on the classes: EXPRESSION, VIRUS, CELL BINDING, and RESPONSE-IMMUNE-GENE-CLASS-CELL (Group 1). The next two profiles contained the classes: AUTOIMMUNE, INFECTED-IMMUNOCOMPROMISED, DEATH-COMMUNICATION, and CYTOTOXIC (Group 2). The last two profiles were created using: REGENERATION, EVOLUTION-DNA, RECEPTOR-ENDOCYTOSIS, and APOPTOSIS (Group 3). Henceforth, we refer to these three groups of profiles, created using the automated process, as AutoProfiles. For each particular profile in the AutoProfiles set, we created an approximately equivalent profile using the corresponding modal MeSH classes. For example, for the last AutoProfiles group, we created a MeSH group containing: REGENERATION, EVOLUTION, ENDOCYTOSIS, and CELL DEATH. Six such profiles, divided into three groups, were created using the MeSH classes. Henceforth, we refer to these three groups of profiles as MeshProfiles. Randomly generated values between 0.2 and 1, were used as the interest values for all the profiles in the AutoProfiles set. The only difference between the pair of profiles in each group, therefore, was their interest values. For each profile in the AutoProfiles set, the corresponding profile in the MeshProfiles set received the same interest values. The three profile groups were selected so that their overall homogeneity varied across groups (in the automated level). In the AutoProfiles set, the Group 1 average homogeneity was 0.24, the Group 2 average homogeneity was 0.65, and the Group 3 average homogeneity was 0.91. The aim here was to see how differences in homogeneity influence filtering at the automated level.

There were 1,440 documents in the test set. Hence, by processing 30 documents per session (ranked and presented), a total of 48 such sessions could be conducted. The AutoProfiles were entered, one at a time, into a version of SIFTER utilizing the automatically produced classification scheme, and for each profile, 48 filtering sessions were executed. Similarly, another version of SIFTER using the MeSH classes was applied, and using the MeshProfiles, one at a time, 48 filtering sessions were completed. For the MeshProfiles and the AutoProfiles, by averaging scores of the six profiles in each set, we derived an average FDS score. These average FDS scores, on a session by session basis, are presented in Figure 2. As can be seen in Figure 2, in almost all the sessions the MeshProfiles outperformed the AutoProfiles. That is, the MeshProfiles succeeded in presenting more relevant documents at the top, ranked between 1–10, than did the AutoProfiles. Overall, the average FDS for the six MeshProfiles was 3.9, whereas the average FDS for the six AutoProfiles was 2.6. In terms of raw FDS scores, the MeshProfiles presented 1,138 documents at the top (ranked between 1–10), whereas AutoProfiles placed 752 documents at the top. The difference, 386 relevant documents, represents a substantial proportion of the total documents that AutoProfiles failed to rank at the top.

Normalized precision (Equation 3) data was also collected for each filtering session, for all the profiles in both the MeshProfiles and the AutoProfiles sets. The normalized precision measure attempts to capture the performance of the filtering system in a more comprehensive way than the FDS score. Whereas the FDS score only shows a raw value expressing the number of documents placed in a narrow range (between 1–10), the normalized precision score summarizes the ranking performance of all the relevant documents present in the incoming stream. To identify broad patterns, the normalized precision score of the six profiles in the MeshProfiles and in the AutoProfiles were averaged for each session. These normalized precision scores are presented, on a session by session basis, in Figure 3, in raw format, and in Figure 4, in smoothed format (five consecutive sessions averaged). During the early period of use, between sessions 1 through 12, filtering performance is highly oscillatory for both MeshProfiles and AutoProfiles. This oscillation is especially clear in the raw format of the normalized precision data. During this latency period, little information exist in the internal user profile, $\hat{d}$, hence, the ranking of documents is generally poor and unpredictable. When filtering performance beyond the latency period was considered, it was found that the MeshProfiles produced superior normalized precision more frequently than the AutoProfiles. The normalized precision of the MeshProfiles beyond latency ranged between 0.61 and 0.93, whereas the normalized precision of the AutoProfiles beyond latency ranged between 0.45 and 0.91. The normalized precision beyond the latency produced by the AutoProfiles continued to show more oscillation (or variability) compared to the normalized precision produced by the MeshProfiles. In other words, performance with the MeshProfiles tended to be more predictable overall than the performance with the AutoProfiles. The smoothed format of the normalized precision data, brings out the general superiority of the MeshProfiles over the AutoProfiles, across all sessions, in a more clear fashion (Fig. 4).

For comparative purposes, we considered the average scores of each group, for both the AutoProfiles and MeshProfiles sets. On a group-by-group basis, the MeshProfiles showed that they were more consistently superior. None of the groups of the AutoProfiles outperformed the corresponding groups in the MeshProfiles, in terms of both the FDS and normalized precision scores. When filtering per-

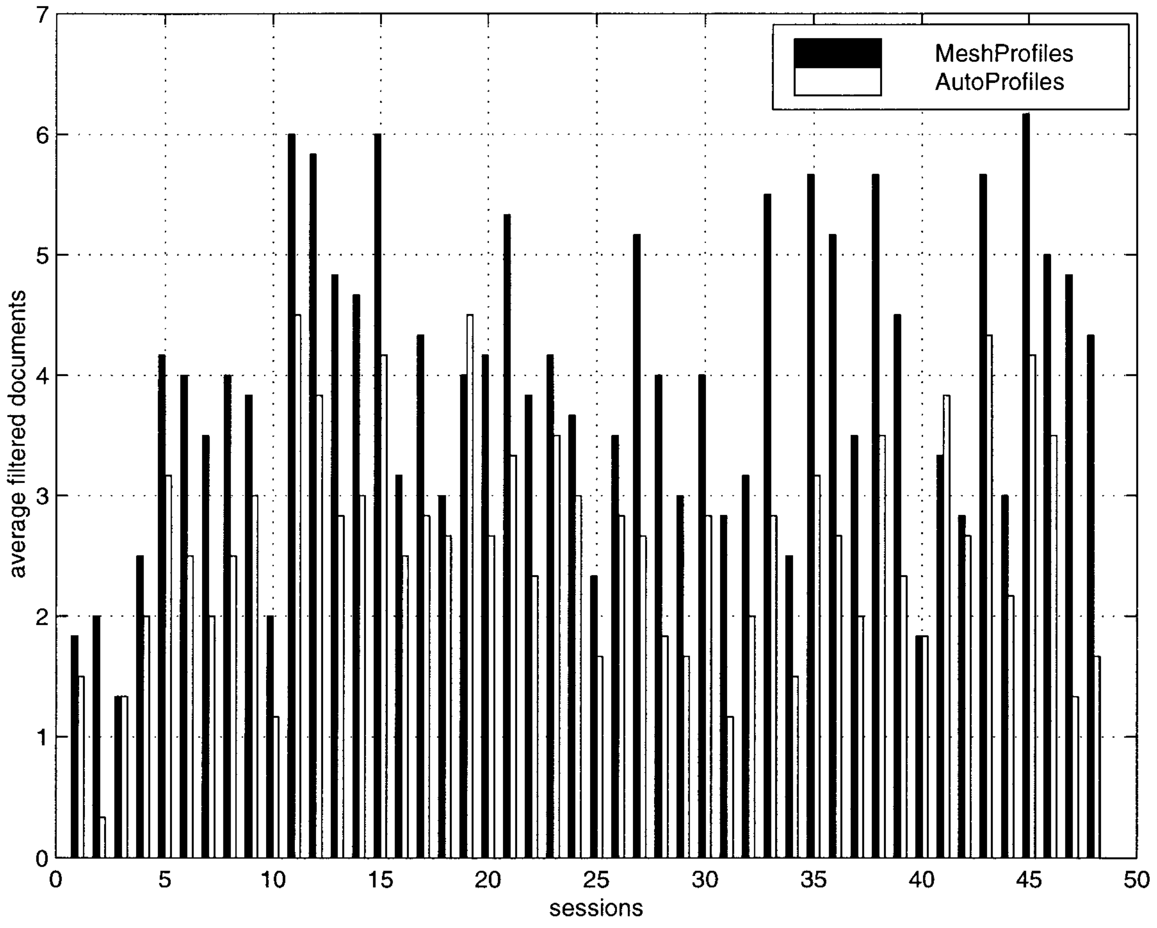

FIG. 2. Average filtered documents of six profiles.

formance was considered, on a group-by-group basis, it was found that the difference in profile content influenced the AutoProfiles set more significantly than the MeshProfiles set. That is, the MeshProfiles performance varied less across groups. The average FDS scores of the MeshProfiles groups ranged between 4.3 and 3.6, and for AutoProfiles they ranged between 3.4 and 1.7. In terms of the raw FDS values, the range of MeshProfiles was a high of 418 (Group 1) and a low of 353 (Group 2). But, the raw FDS values of the AutoProfiles ranged between a high of 331 (Group 1) and a low of 171 (Group 2). This variation pattern generally held true when the normalized precision data was considered. The MeshProfiles average normalized precision ranged between 0.84 (Group 1) to 0.71 (Group 3). Whereas the average normalized precision of the AutoProfiles ranged between 0.78 (Group 1) to 0.58 (Group 2). To determine if the higher variability in the AutoProfiles set had any logical pattern, we compared session-by-session performance (average FDS scores) of each group to the corresponding group in the MeshProfiles set using Pearson's Correlation Coefficient (r). We found that AutoProfiles-Group 1 (low homogeneity) had a r of 0.31 when compared to the MeshProfiles-Group 1 FDS scores. The AutoProfiles-Group 2 (medium homogeneity) had a r of 0.54 when compared to the MeshProfiles-Group 2 FDS scores. The r was 0.79 when the FDS scores of the AutoProfiles-Group 3 (high homogeneity) and the MeshProfiles-Group 3 were compared. All these correlations were found to be significant at the 0.05 level. These r results suggested that with increasing homogeneity in the AutoProfiles set, performance tended to stabilize and become more similar to the control level.

Discussion

Overall, the MeshProfiles performed superior filtering as compared to the AutoProfiles. Generally, AutoProfiles failed to place as many relevant documents in the top 10 as the MeshProfiles. The primary contributing factor that we identified was the dispersion of documents into multiple automatically produced classes. There were more automatically produced classes than the actual MeSH classes, and they varied according to their homogeneity. The dispersion of documents in the automatic level lead to net reduction in number of documents that were ultimately classified into classes flagged as relevant in the profiles. Conversely, two factors increased the likelihood of the control level producing higher FDS score. One, documents belonging to relevant classes were maintained together in the control level

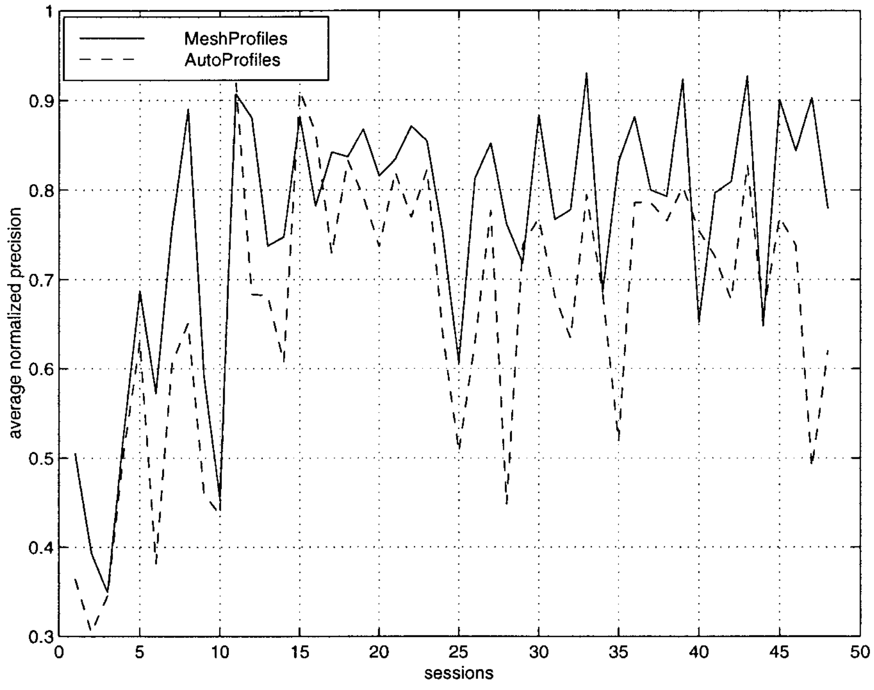

FIG. 3. Average normalized precision of six profiles.

(as these classes were, of course, more homogeneous). Two, there were fewer classes in the control level than in the treatment level, therefore, there were more documents per class.

The difference in normalized precision scores was less prominent, although these scores generally were higher for MeshProfiles than AutoProfiles. Overall, when MeshProfiles were used, more relevant documents were identified and, consequently, more documents had to be ranked accurately. Conversely, the AutoProfiles identified fewer relevant documents per session and had a less difficult task of ranking. This is probably a strong contributing factor in the superior, yet smaller, difference we found in the normalized precision scores.

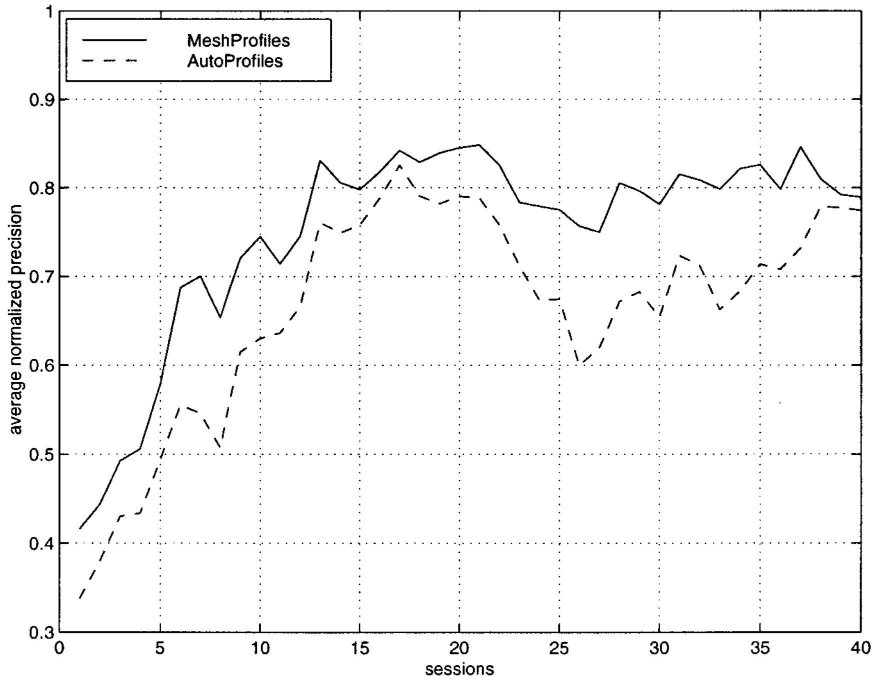

FIG. 4. Average normalized precision of six profiles in smoothed format.

When the classification component is considered, independent of filtering, one particular advantage of automatic classification becomes apparent. In this research, we have used a training document set to automatically generate a thesaurus and, subsequently, a classification scheme. As an outcome of this process, we have found some new concepts that were highly similar (a precise degree of similarity was also found) to concepts in MeSH, but which did not actually appear in the original MeSH set. This points toward a potential application of the automated classification mode, on carefully selected document sets, for automatic term or concept harvesting—the results of which could be used to enhance an existing classification scheme.

Human classification can add value to the filtering process in terms of improved reliability, but it can be labor intensive. Automated classification produces degraded filtering, but it requires little or no human intervention. The architecture of the SIFTER system distributes the overall filtering effort in a manner that permits both human and automated input. In our view, the two approaches can complement each other, and a balance between them may produce better results than exclusive dependence on one approach. We are, however, at an early stage of experience with SIFTER. Considering the numerous factors that can influence filtering, further studies are needed to clearly establish the "trade-off" points between the two approaches.

Conclusion and Future Work

The research reported here focused on analyzing classification performance in the context of filtering. A methodology that utilizes human classification as a control for measuring automated classification performance was presented. The automated classification approach was implemented using two different algorithms. A vocabulary discovery algorithm was developed to identify prospective terms for a thesaurus. An unsupervised cluster discovery algorithm called the *Maximin-Distance* algorithm was used for classification scheme generation. Analysis of automated classification performance showed that the top 14 automatically produced classes had homogeneity levels higher than 50%, and 57% of the documents were classified into these classes. Using the classification performance of individual classes as a basis, we then created several interest profiles. Subsequently, two versions of the filtering system, one using the automatically produced classification scheme and one using MeSH classes, were employed to perform filtering while keeping the user profiles constant across the two levels of classification. The filtering results showed that with degradation in classification homogeneity, the filtering performance may also degrade, and with high homogeneity (human level), superior filtering performance can be attained.

We are presently attempting to extend this work in three specific ways. One area involves integrating direct user interaction into the classification process. We plan to extend SIFTER's user interface to allow users to provide explicit classificatory input, such as new terms for the thesaurus or modifications to the structure of the thesaurus and the classification scheme. A second area involves applying a human produced classification scheme as the core scheme, but supplementing and enriching such a scheme in an ongoing fashion through automated means. The third area involves clarifying classification performance and its effect on filtering based on varying document formats. We wish to consider a range of such formats, including full text and HTML.

Acknowledgments

The Purdue School of Science Computer Science Department and the Indiana University School of Library and Information Science supported the SIFTER project from its inception. Our thanks go to the other members of the SIFTER research team, specifically Snehasis Mukhopadhyay and Wai Lam, for vigorous discussions and for aiding in the software development. We are grateful to the anonymous referees for their valuable comments. We also offer our thanks to Sigma Mostafa. Her background in cell biology was invaluable in expanding our understanding of the domain and in refining the data analysis.

References

Allan, J. (1996). Incremental relevance feedback for information filtering. In H. Frei, D. Harman, P. Schauble, & R. Wilkinson (Eds.), *Proceedings of 19th ACM International Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, August 18–22, 1996 (pp. 270–278). New York: ACM.

Arnheim, L. (1996). Summary of proceedings. *Workshop on Collaborative Filtering*, Berkeley, CA. Available: http://www.sims.berkeley.edu/resources/collab/collab-report.html

Atkins, D. E., Birmingham, W. P., Durfee, E. H., Glover, E. J., Mullen, T., Rundensteiner, E., Soloway, E., Vidal, J. M., Wallace, R., & Wellman, M. P. (1996). Toward enquiry-based education through interacting software agents. *IEEE Computer Theme Issue on Digital Library Initiatives*, *29*(5), 69–77.

Beaulieu, M. M., Gatford, M., Xiangji, H., Robertson, S. E., Walker, S., & Williams, P. (1997). Okapi at TREC-5. *Proceedings of the Fifth Text Retrieval Conference*, Gaithersburg, MD. Available: http://www.nlpir.nist.gov/TREC/

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

Boughanem, M., & Soule-Dupuy, C. (1997). MercureO2: Adhoc and routing tasks. In E. M. Voorhees & D. Harman (Eds.) *Proceedings of the Fifth Text Retrieval Conference*, Gaithersburg, MD. November 20–22, 1996. Available: http://www-nlpir.nist.gov/TREC/

Brajnik, G., Guida, G., & Tasso, C. (1990). User models in expert man-machine interface: A case study in intelligent information retrieval. *IEEE Transactions on Systems, Man and Cybernetics*, *20*(1), 166–185.

Cheng, P. T. K., & Wu, A. K. W. (1995). ACS: An automatic classification system. *Journal of Information Science*, *21*(4), 289–299.

Etzioni, O., & Weld, D. (1994). A Softbot-based interface to the Internet. *Communications of the ACM*, *37*(7), 72–76.

Fischer, G., & Stevens, C. (1991). Information access in complex, poorly structured information spaces. In S. P. Robertson, G. M. Olson, & J. S. Olson (Eds.), *Proceedings of ACM Special Interest Group on Computer*

Human Interaction Annual Conference New Orleans, LA, April 27–May 2, 1991 (pp. 63–70). New York: ACM.

Jacob, E. K. (1991). Classification and categorization: Drawing the line. In B. H. Kwasnik & R. Fidel (Eds.), *Advances in classification research* (*Vol. 2*, pp. 67–83). Washington DC: American Society for Information Science.

Jacobs, P. S., & Rau, L. F. (1990). SCISOR: Extracting information from on-line news. *Communications of the ACM, 33*(11), 88–97.

Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., & Riedl, J. (1997). GroupLens: Applying collaborative filtering to Usenet News. *Communications of the ACM, 40*(3), 77–87.

Lam, W., Mukhopadhyay, S., Mostafa, J., & Palakal, M. (1996). Detection of shifts in user interface for personalized information filtering. In H. Frei, D. Harman, P. Schauble, & R. Wilkinson (Eds.), *Proceedings of 19th ACM International Conference on Research and Development in Information Retrieval,* Zurich, Switzerland, August 18–22, 1996 (pp. 317–325). New York: ACM.

Lang, K. (1995). *NewsWeeder: An adaptive multi-user text filter* (Tech. Rep.). Pittsburgh, PA: Carnegie-Mellon University, School of Computer Science.

Larson, R. (1992). Experiments in automatic Library of Congress classification. *Journal of the American Society for Information Science, 43,* 130–148.

Lewis, D. D. (1992). Text representation for intelligent text retrieval: A classification-oriented view. In P. S. Jacobs (Ed.), *Text-based intelligent systems: Current research and practice in information extraction and retrieval* (pp. 179–197). Hillsdale, NJ: Erlbaum.

Maes, P. (1994). Agents that reduce work and information overload. *Communications of the ACM, 37*(7), 31–40.

May, A. D. (1997). Automatic classification of e-mail messages by message type. *Journal of the American Society for Information Science, 48,* 32–39.

Mostafa, J., Mukhopadhyay, S., Lam, W., & Palakal, M. (1997). A multilevel approach to intelligent information filtering: Model, system, and evaluation. *ACM Transactions on Information Systems, 15*(4), 368–399.

Mukhopadhyay, S., Mostafa, J., Palakal, M., Lam, W., Xue, L., & Hudli, A. (1996). An adaptive multi-level information filtering system. In D. Chin (Ed.), *Proceedings of the 5th International Conference on User Modeling* (pp. 21–28). Kailua-Kona, HI: UM Inc.

Narendra, K. S., & Thathachar, M. A. L. (1989). *Learning automata—an introduction.* Englewood Cliffs, NJ: Prentice-Hall.

Oard, D. W. (1997). *Information filtering resources.* Available: http://www.ee.umd.edu/medlab/filter

Oard, D. W., & Marchionini, G. (1997). *A conceptual framework for text filtering* (Tech. Rep. No. 3643). College Park, MD: University of Maryland, Department of Computer Science.

Rich, E. (1983). Users are individuals and individualizing user models. *International Journal of Man–Machine Studies, 18,* 199–214.

Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval.* New York: McGraw-Hill.

Shapira, B., Shoval, P., & Hanani, U. (1997). Stereotypes in information filtering systems. *Information Processing & Management Systems, 33*(3), 273–287.

Crawford, D. (Ed.). (1997). Recommender systems—Linking users by similar interests [Special issue]. *Communications of the ACM, 40*(3).

Tou, J. T., & Gonzalez, R. C. (1974). *Pattern recognition principles.* Reading, MA: Addison-Wesley.

Yang, Y., & Chute, C. (1994). An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems, 12*(3), 252–277.