

Effects of prior knowledge on the effectiveness of a hybrid user model for information retrieval

Hien Nguyen¹ Eugene Santos Jr.²

¹ Department of Mathematical and Computer Sciences
University of Wisconsin-Whitewater
800 W. Main Street. Whitewater, WI 53190
nguyenh@uw.edu

² Thayer School of Engineering
Dartmouth College
8000 Cummings Hall. Hanover, NH 03755
Eugene.Santos.Jr@Dartmouth.EDU

ABSTRACT

To quickly find relevant information from huge amounts of data is a very challenging issue for intelligence analysts. Most employ their prior domain knowledge to improve their process of finding relevant information. In this paper, we explore the influences of a user's prior domain knowledge on the effectiveness of an information seeking task by using seed user models in an enhanced information retrieval system. In our approach, a user model is created to capture a user's intent in an information seeking task. The captured user intent is then integrated with the attributes describing an information retrieval system in a decision theoretic framework. Our test bed consists of two benchmark collections from the information retrieval community: MEDLINE and CACM. We divide each query set from a collection into two subsets: training set and testing set. We use three different approaches to selecting the queries for the training set: (1) the queries generating large domain knowledge, (2) the queries relating to many other queries, and (3) a mixture of (1) and (2). Each seed user model is created by running our enhanced information retrieval system through such a training set. We assess the effects of having more domain knowledge, or more relevant domain knowledge, or a mixture of both on the effectiveness of a user in an information seeking task.

Keywords: user model, hybrid user model, personalized information retrieval, relevance feedback, prior domain knowledge, evaluation.

1. INTRODUCTION

One of the challenging issues encountered every day by intelligence analysts is the need to quickly find relevant information from a massive amount of data. In order to accomplish their tasks, they must use their domain knowledge to search for information. By domain knowledge, we refer to a user's understanding and knowledge of a specific topic. Domain knowledge has been found to affect a user's performance in search^{1,24} and affect a user's hypermedia learning¹². For example, analysts with more knowledge of a specific topic may know the right terms to retrieve the relevant information to solve the task at hand. They may also better know who to contact and where to find that needed information at the right time. As well, they may know how to use their knowledge to present their findings within the bigger picture at hand.

Domain knowledge is one of the factors used to differentiate between novice and expert users for a specific application. One of the misleading impressions regarding the differences between novice and experienced users is that the more experienced users would have more domain knowledge which leads to more successful retrieval tasks. Many intelligent information retrieval applications have been trying to use and capture a user's domain knowledge to help him/her to retrieve more relevant documents quickly (for example: ^{2,21,8,17,15}). However, while there are many research projects

focused on how to construct and use domain knowledge in an information retrieval application, very little effort has been paid to quantitatively evaluating how the captured user's domain knowledge affects his/her effectiveness.

In our approach, a user model is created to capture a user's intent in an information seeking task. The captured user intent is then integrated with the attributes describing an information retrieval system through a decision theoretic framework. One of the key features of our approach is that we capture a user's domain knowledge dynamically by analyzing the commonality of the retrieved relevant documents (relevance as indicated by the user). In this paper, we are interested in addressing an important research question:

“Does more general domain knowledge which is captured by a user model, lead to a more effective search? If not, what kind of domain knowledge would potentially lead to a more effective search?”

This problem is critical because it helps to shed light on how to effectively capture and use domain knowledge to improve a user's effectiveness in an information seeking task. Additionally, it helps in the future design of more efficient intelligent information retrieval applications. However, it is a challenging problem because it involves two open research issues: (1) How to dynamically capture a user's domain knowledge, and (2) how to assess the knowledge needed for improving the effectiveness of an information retrieval application. Furthermore, there are currently few or no standard procedures for evaluating a user's domain knowledge making it even more difficult. In this paper, we present our study addressing the above research question and an evaluation approach that takes into account both open issues.

In our evaluation, we use two benchmark collections from the information retrieval community: MEDLINE and CACM. We divide each query set from a collection into two subsets: training set and testing set. We use three different approaches to select the queries to form a training set for each collection: (1) the queries that generate a large domain knowledge, (2) the queries that are related to many other queries in a testbed and (3) a mixture of both (1) and (2). Each seed user model is created by running our enhanced information retrieval system through such a training set in order to capture the notion of prior domain knowledge. We assess the effects of having more general domain knowledge, or more relevant domain knowledge to the tasks at hand, or a mixture of both on the effectiveness of a user in an information seeking task. We hypothesize that having more relevant domain knowledge to the task at hand would be crucial to improve the effectiveness of a user's performance in an information seeking task. The main reason behind this hypothesis is that larger domain knowledge may not be enough to a successful retrieval task because if this domain knowledge centered around a broad topic that subsumes the topics of interests by the users, they may choose to issue the queries by using the terminologies loosely relating to the task. That may lead to a worse retrieval performance. Therefore, the domain knowledge that closely relates to the task at hand is crucial to helping users to retrieve more relevant information.

The study presented in this paper evaluates the knowledge captured by a computational model in an enhanced information retrieval application. We focus on finding the answer or supported evidence to whether having more domain knowledge leads to more effectiveness in a retrieval task. The challenges of our study include the difficulty in capturing a user's real knowledge using a computational model and the lack of instruments to evaluate how useful it is. The significance of our result is two fold: First, it sheds some lights on the usefulness of the quality and quantity of captured knowledge. Secondly, it helps in designing an intelligent information retrieval application with regards to what kind of knowledge we should focus.

This paper is organized as follows: We begin with an overview of related work followed by a presentation of our hybrid user model. Next, the evaluation procedure and testbed are discussed. Our results are presented and we conclude with future work.

2. RELATED WORK

In this section, we present existing work that directly or indirectly evaluates the effect of domain knowledge on information seeking. By direct evaluation, we refer to the studies that assess the effect of a real user's domain knowledge on his/her searching behavior or retrieval success. In an indirect evaluation, we assess a user's domain knowledge captured by a computational model instead.

In the information retrieval community, domain knowledge affects many stages in an information seeking task from query formulation to selecting, seeking, and presenting the relevant information. In the query formulation process, users with more experience in a search domain may know the terminologies that are used frequently in the domain, which may be helpful in the process of identifying relevant documents more quickly. Terms and their relationships are used in formulating the right queries and in developing the users' search tactics to find the relevant concepts of the queries quickly. In the early 90s, when user models were first used to improve the user's performance in information retrieval, Allen¹ found that more experienced users tend to expand the search more than novice users. The total amount of time preparing for the search and the number of queries in a search have been found to be influenced by the level of domain knowledge⁷. Studies such as^{22,24} have shown that users with higher level of domain knowledge may issue more queries and use more words in their queries to find relevant information, while the average number of terms used in a query also increased. A study conducted by Wildmuth²³ shows that low level of domain knowledge is associated with less efficient choices for terms being included in a search and thus prevents users from quickly reaching their final searching goal. All these studies mentioned above directly assess the effect of a user's domain knowledge on search.

Even though many studies have found that domain knowledge positively affected the user's behaviors in an information seeking task such as by, increasing the number of terms and the number of queries, very few studies have explored the relationship between the level of domain knowledge and the amount of retrieved relevant information. Most of the research in this direction from the early 90s to present have unsuccessfully found any cause and effect relationship or any relationships between the level of domain knowledge and retrieval success^{19,23}.

While many intelligent information retrieval applications have evaluated the effect of a user's domain knowledge on retrieval success in general, to the best of our knowledge, there are only a few studies that focus on analyzing what kind of domain knowledge most affects the user's effectiveness in an information seeking task (for example: a study that evaluates the use of knowledge of a search topic or knowledge of a search system¹²). In this paper, we focus on the assessment of effects of general domain knowledge and knowledge relates to the task at hand on the retrieval success.

3. HYBRID USER MODEL

In this section, we provide background on our user modeling approach which is used to improve the effectiveness of a user engaged in an information seeking task by building a model about him/her that integrates information from both the user side and system side in a decision theoretic framework. We need user models for information retrieval applications because the traditional information retrieval framework does not take into account much of the inputs available from a user except for queries and relevance feedback. We start by describing the process of determining information about a user and then present how we combine this information with components of an information retrieval (IR) application in a decision theoretic framework. We focus on the user side first because the information about a traditional IR application is usually determined while the system is being developed and used.

3.1 User intent

Our goal is to capture user intent in information seeking. In our approach, a user's intent is partitioned into three components: Interests, Preferences, and Context. The Interests refers to what users are doing to achieve their goals, the Preferences captures how the users may go about achieving their goals, and the Context infers why they are trying to achieve these goals. We capture the Context, the Interests, and the Preferences aspects of a user's model with a context network (C), an interest set (I), and a preference network (P)^{17,15,14,13}.

A context network (C) is a directed acyclic graph (DAG) that consists of concept nodes and relation nodes. Concept nodes represent concepts in noun phrases. The relation nodes are divided into two categories: "isa" representing "set-subset" relationships among concepts and "related-to" relations representing the general relations between one concept to another one. Our Context network is created dynamically by finding a set of sub-graphs in the intersection of all retrieved relevant documents. Each document is represented as a document graph (DG) which is similar to the representation of a context network. For each document, we construct its corresponding DG by first automatically extracting noun phrases (NPs) from natural language text using Link Parser²⁰. We then use three heuristic rules to construct the concept nodes and relation nodes: *noun phrase heuristic*, *noun-phrase preposition phrase heuristic*, and *sentence heuristic*. The main idea of these rules is to establish the relationship among noun phrases by exploiting

syntactic structure of a noun-phrase, a preposition phrase, or a sentence. More details on DG construction can be found in ²⁵.

The Interests are captured in the interest set which consists of current concepts that a user is focusing on and how much emphasis he/she has placed on them. The interest set is initially determined from the current query, and the set of common sub-graphs. The Preference network is captured in a Bayesian network ⁹. There are three kinds of nodes in this Preference network: pre-condition (Pr), goals(G), and action nodes(A). Each node has two states: true and false. Precondition nodes represent the requirements to achieve the goal nodes. Goal nodes represent the tools that are used to modify a user's query. We currently have the two tools: (i) filter which narrows down a query semantically, and (ii) expander which "broadens" up a query semantically. The conditional probability table of each goal node is similar to the truth table for logical AND. Each goal node is associated with only one action node. The pre-condition nodes in this example consist of interest concepts and current query nodes. These nodes will be set as evidence (i.e., to *true*) if they belong to the current interest set or are fully/partially matched with the current query. The filter or expander nodes simply mean that the action node associating with them will contain a link to a *query sub-graph* that is narrower or broader than the original query sub-graph. A query graph is similar to the document graph but it is generated from a user's keyword/text query instead of a full text document. The Preference network is updated when a user gives feedback. Basically, we add to the current Preference network the tool that helps most in the previous retrieval processes. If the total number of retrieved relevant documents exceeds a user-defined threshold, a tool is considered helpful. For more information, please see ^{14,13}.

In essence, Context network contains information used to modify a query's graph while Interest set and Preference network contain information used to determine how we modify the query graph. The process of modifying a user's query graph is shown as follows:

Step 1: We are given a user model $M=\{I, P, C\}$ and a query graph (QG) q .

Step 2: We set as evidence all interest concepts found in P . We find all pre-condition nodes (Pr) representing a query in P which has associated query graph (QG) that completely or partially matches against the given q . If such a node is found, it is set as evidence.

Step 3: Belief updating on P is performed. We choose the top n goal nodes from P with highest probability values (G).

For every goal node g in G :

If the query has been previously submitted and the user has used this goal, the original query sub-graph is replaced by the graph associated with the action node of this goal.

If the query has not been asked before and g represents a filter: For every concept node q_i in the user's query graph q , we search for its corresponding node cq_i in C . For every concept a_i in I , we search for its corresponding node ca_i in C such that ca_i is an ancestor of cq_i . If such c_i and cq_i are found, we add the paths from C between these two nodes to the modified query graph. It works similarly with an expander except that ca_i should be a progeny of cq_i .

Step 4: The modified QG (q) is sent to the search module where it is matched against each DG representing a record in our database. Those records that have the number of matches greater than a user-defined threshold are chosen and displayed to a user. A match between a QG and a DG is defined as

$$sim(q, d) = \frac{n}{2 * N} + \frac{m}{2 * M}$$

where n, m are the number of concepts and relation nodes of q found in d respectively. N, M are the total number of concept and relation nodes of q . Two relation nodes are matched if and only if at least one of their parents and one of their children are matched. For more details about our approach, please see our earlier papers ^(17,15,14).

3.2 Combining user intent with elements of an IR system

We combine the user intent captured in Section 3.1 with the elements of an IR application in a decision theoretic framework. Typically, an IR system consists of a *query, indexing scheme, similarity measure, threshold, and collection*. Query represents a user's request. Indexing schemes contain a hierarchy of related terms. A similarity measure is a function which determines how similar a user's query and a document from the searched collection is. Threshold is a real number which indicates how we should filter out irrelevant documents. A collection usually consists of a set of documents in a specific topic such as aerodynamics or medicine.

A user's effectiveness is determined by using a function called the *effectiveness function* F_e . An example of such a function can be precision, which is the ratio of the number of retrieved relevant document over the number of retrieved documents. Another example of F_e is recall which is the ratio of the number of retrieved relevant documents over the number of all relevant documents. Unfortunately, the computation of F_e is post-retrieval. This means that we only have enough information to compute the effectiveness of a search after a user has issued a query to an IR system and the IR system has returned a set of documents to the user. In this hybrid user model, we use a pre-retrieval mechanism to estimate F_e and we choose the solution that will likely improve F_e in the future retrievals.

Our solution is to convert this problem into a multi-attribute decision problem and use multi-attribute utility theory¹⁰ in which a set of attributes is constructed by combining the set of attributes describing a user's intent and the set of attributes describing an IR system. In multi-attribute utility theory, the decision is made based on the evaluation of *outcomes* of the actions performed by a user or an agent. In this problem, the outcome space represents the set of *all* possible combinations of information about a user and information about a system. We use multi-attribute utility theory because the estimation of the effectiveness function with respect to the searching goal is a problem of preference elicitation because it represents a user's preferences over a space of possible sets of values describing a user and describing an IR system. We use the research on predicting query performance in IR and computing dissemination threshold in information filtering (IF) in our elicitation technique.

In this hybrid user model, we focus on only two attributes: query (Q) and threshold (T) because the other attributes such as collections, similarity measure, and indexing schema are determined when we build an IR system. Additionally, the query attribute subsumes the attributes about a user's interests, preferences and context because interests, preferences and context are used to modify a user's query as described in Section 3.1. Thus, they do not participate in the decision making process. We evaluate each outcome with a real-valued function. We make another assumption here that these two attributes are preferentially independent. Thus, the value function representing a user's preferences over these two attributes can be constructed as follows:

$$V(Q,T) = \lambda_1 V_1(Q) + \lambda_2 V_2(T)$$

where λ_i represents the importance of attribute i to the user, and V_i is a sub-value function for the attribute i ($i=1$ or $i=2$). This value function is generic for all IR systems and all type of users. As it is very difficult to elicit the coefficients λ_i , we determine the partial value function V_i and follow this rule:

"The partial value function implies that an outcome x_1 with the value x_{11}, x_{12} is preferred to an outcome x_2 with value x_{21}, x_{22} if and only if

$$\begin{aligned} x_{1i} &\leq x_{2i} && \text{for } i=1,2, \text{ and} \\ x_{1i} &> x_{2i} && \text{for some } i." \end{aligned}$$

For each sub-value function, each attribute needs to be evaluated with respect to a user's effectiveness in achieving a searching goal at a given time. We assume that a user's searching goal at any given time is to retrieve many relevant documents quickly for the user. Therefore, we choose the average precision at fixed point recalls as the effectiveness function because it measures both the percentage of retrieved relevant documents and the speed of retrieving these documents. This function takes an average of precision at three points of recall (recall=0.25, 0.5, and 0.75).

Sub-Value Function over Query

We have chosen the standard deviation of a query's terms' *inverted document frequency* (*idf*) as the core of this sub-value function. The main idea of *idf* measure is that the less frequent terms in a collection are the terms with more discriminating power. The main reasons for our choice of *idf* are that (i) it has shown relatively good positive correlation with the average precision metric⁶, and (ii) it is a pre-retrieval measure. Recall that each query in this approach is represented by a query graph¹⁷. Each query graph contains concept node and relation nodes. Therefore, we define the sub-value functions for the concept nodes and for the relations. The sub-value function for the concept nodes is defined as follows:

$$V_c(q) = \sqrt{\frac{1}{n} \sum_{c \in q} (idf(c) - \mu_{idf}(q))^2}$$

$$\mu_{idf_c}(q) = \sum_{c \in q} \frac{idf(c)}{n} \text{ and } idf(c) = \frac{\log_2(N+0.5)/N_c}{\log_2(N+1)}$$

where n is the number of concept in query q , N is the total number of documents in a collection and N_c is the total number of documents containing the concept c .

The sub-value function for the relation nodes is defined in a similar manner. For more detail, please see ^{13,14}.

Sub-Value Function for Threshold

We take advantage of research from adaptive threshold in information filtering to construct a sub-value function for thresholds. We choose the threshold of the last document seen by a user and the percentage of returned documents preferred to be seen by a user as the core of our sub-value function as it is done in ⁴. The intuition behind this approach is that if the number of retrieved relevant documents is small, and the difference between the similarity of the last returned documents and the threshold is big, then we need to decrease the threshold considerably in order to retrieve more relevant documents.

The sub-value function for the threshold is defined as follows:

$$V(T) = \begin{cases} 1 & \text{if } T > T_t \\ 0 & \text{otherwise} \end{cases} \text{ with } T_{(t+1)} = T_t + \frac{sim(d_{last}) - T_t}{e^{\frac{(R_t - \lambda)}{\phi}}}$$

where $\lambda = 1300$ and $\phi = 500$, R_t is the total number of relevant document at time t , and d_{last} is the similarity of the last retrieved document in the previous retrieval. This method of updating threshold is chosen because it is light-weight and can be computed in the pre-retrieval process. It also has been shown to correlate well with average precision in ⁴. Please see ¹³ for more details.

3.3 Complexity and Implementation of Hybrid User Model

The process of computing idf for every concept and every relation can be done offline in polynomial time because in a traditional information retrieval application, the document collection is static. The complexity of this process is $O(nm)$ with n being the number of documents and m being the maximum number of nodes in a document graph. The only online algorithms are the computation of sub-value functions over a query and over a threshold for those concepts and relations included in a user's query.

The computation of a sub-value function over a query for concept nodes has complexity of $O(l_c \log_2(N) + l_c)$ with l_c being the number of concepts in a query and N being the number of concepts in the collection. Similarly, the computation of a sub-value function has complexity of $O(l_r \log_2(N) + l_r)$ with l_r being the number of relations in a query, and N being the number of relations in the collection.

3.4 Implementation:

After the user issues a query q , this query is modified using the information in the Interests, Preferences and Context as described in Section 3.1. Assuming that there are m goals fired in the Preference network, each goal generates a query $\{q_1, q_2, \dots, q_m\}$. We then use sub value functions to evaluate each q_i and we choose the query with highest sub-value function evaluation. We ask the user directly about his/her preference of threshold (T_0). We send the query with highest value evaluated above to the search and filter out the retrieved result based on the value of the threshold. We may update $V(T)$ if needed.

4.EVALUATION

4.1. Testbed:

MEDLINE and CACM are chosen as our testbeds in this evaluation. We chose these collections because they have been used widely in the IR community to evaluate the effectiveness of relevance feedback techniques ^{16,11,5}. In particular, CACM contains 3204 documents and 64 queries in computer science and engineering (CSE); while MEDLINE contains 1033 documents and 30 queries in the medical domain. Each query in a collection also associates with a list of relevant documents already determined by domain experts.

4.2 Construct seed models:

We create a *seed* user model by starting with an empty model and run our program through each *training set*. We divide the queries for each collection into two sets: a training set that contains one third of the number of queries and a test set that contains the remaining queries. We chose three ways to construct the training set.

The first way is to choose the queries generating large context networks. We obtain a list of context networks that are constructed from running each query twice starting from an empty user model in ¹³. Next, we choose the queries corresponding to the top third of largest context networks to be included in the training set. For MEDLINE collection, the biggest size for a context network generated by a query in the training set is 62K while it's 40K for CACM. The second way is to choose the queries that are related to most of the other queries. For each query, we compute the number of other queries that have non-zero similarities with it. The similarity of any two queries is defined as follows:

$$sim(q_i, q_j) = \frac{c}{c_i + c_j} + \frac{r}{r_i + r_j}$$

where c is the number of concepts nodes shared by both q_i and q_j while r is the number of relation nodes shared by both q_i and q_j . c_i, c_j, r_i, r_j are the number of concept and relation nodes of q_i and q_j , respectively. If two queries have the same number of queries with non-zero similarities, then the one with bigger *averageSim* will be ranked first. *averageSim* is defined as follows:

$$averageSim(q_i) = \frac{1}{n} \sum_{j=1-n} sim(q_i, q_j)$$

where n is the number of queries that have non-zero similarities with query q_i

The queries corresponding to the top third of the list ranked based on the number of queries with which it has non-zero similarities are included in the training set. In the MEDLINE collection, the highest number of other related queries is 13 queries while it is 39 queries for CACM. Note that there is 1 query belonging to both the training set chosen by the first approach and second approach for MEDLINE while there are such 4 queries for CACM.

The third way is to choose half of the queries following the first method and the other half of queries following the second method. The intuition behind these three methods is that we would like to see if more general domain knowledge (first method) or more relevant knowledge to the task at hand (second method), or the mixture of both improves a user's effectiveness in an information seeking tasks. We run our system starting from an empty user model with each training set. We obtain three seed models corresponding to the three above mentioned methods of choosing training sets and call them as *Seed_{knowledge}*, *Seed_{relevant}*, and *Seed_{hybrid}* respectively.

4.3 Procedure to assess the effect of having prior domain knowledge combined with short-term knowledge.

Short-term knowledge refers to the knowledge learned after the system runs each query in the test set. In this procedure, we then start our system with each seed user model. With each query, the relevant documents from the first 15 returned documents are identified from the information on relevant documents to this particular query provided in each collection. These relevant documents are then used to construct a user model to modify the query proactively. We add the concept and relation nodes to the original QG based on the procedure described in previous sections. We choose to use the standard deviation of *inverted document frequency (idf)* over concept nodes in a query as a sub-value function for the query because it is simple and easy to implement. In our preliminary evaluation of several value functions ¹⁴, there is insufficient evidence to support the use of one sub-value function over the others. One good implication from this finding is that we can use a simple sub-value function. We then run each system again with the modified query. We call the first run as *initial run* and the second run as *feedback run*. For each query, we compute average precision at three point fixed recall. After each query, the user model is reset to these seed models, respectively.

4.4 Procedure to assess the effect of having prior domain knowledge combined with long-term knowledge

This procedure is similar to the process of assessing the effect of having prior domain knowledge combined with short-term knowledge described above. However, when we start with a seed user model, we keep update this model without having to reset it. The *long-term* knowledge here is learned from a set of queries as a user goes through each test set.

4.5 Comparing Against a Baseline – TFIDF/Ide dec-hi

We compare our approach against Ide dec-hi with term frequency inverted document frequency (TFIDF) weight¹⁶. The main idea of the Ide dec-hi approach is that the weight of each word in the original query is re-computed using its weights in relevant documents and the first irrelevant document. The words with the highest weights from relevant documents are also added to the original query.

5. RESULTS AND DISCUSSION

The results of our procedure to assess the effect of prior domain knowledge with short-term knowledge is shown in Figure 1 (a,b,c) and the results obtained from assessing the effect of prior domain knowledge with long-term knowledge is shown in Figure 2 (a,b,c). For all seed models, with both short-term and long-term knowledge, our hybrid approach has performed better than the TFIDF/Ide dec-hi approach while being competitive in the feedback runs. This implies that by using the seed models, we can retrieve more quality documents earlier than the traditional TFIDF approach and working competitively with the Ide-dec hi approach in the feedback run. For example: for MEDLINE collection, the average improvement between the average precisions produced by our hybrid user models with all three seed models and produced by the TFIDF approach is 9.9% for seed models with short-term knowledge. We define this improvement by taking the difference between the average precisions produced by hybrid approach and the ones produced by TFIDF approach. This result is also in line with our other results reported in¹³ where we use the whole set of queries in the testbed instead of a subset as it's used in this experiment.

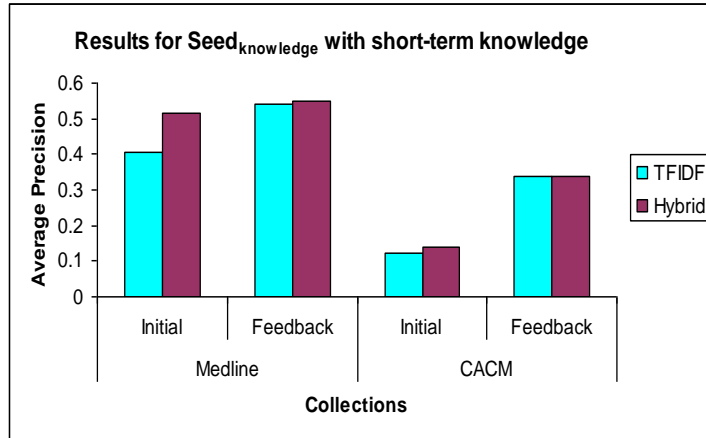
When comparing the improvement obtained from using these three seed models, we note that there is no evidence that strongly supports that one seed is better than the other even though the $Seed_{hybrid}$ seems to produce the least improvement in terms of average precision compared to other two seed models. For example, the improvement between the average precision for using $Seed_{knowledge}$, $Seed_{relevant}$ and $Seed_{hybrid}$ with short-term knowledge are 11.25%, 11.86%, and 6.6% for Medline collection in the initial run. Similarly, the differences between the average precision for using these three seeds with short-term knowledge are 1.7%, 1.9%, and 1.2% for CACM collection in the initial run, respectively.

The main reason is that the sets of queries for both MEDLINE and CACM contain many topics and thus, these queries change rapidly from one subject to the next. For example: Medline contains 30 queries for various topics ranging from kidney to lung to heart disease to blood disease and so forth. Similarly, CACM contains 64 queries about different topics such as Artificial intelligence, Human Computer Interaction, Information Retrieval, Algorithms and Data Structure, Parallel Computing, Distributed Computing, and Operating Systems.

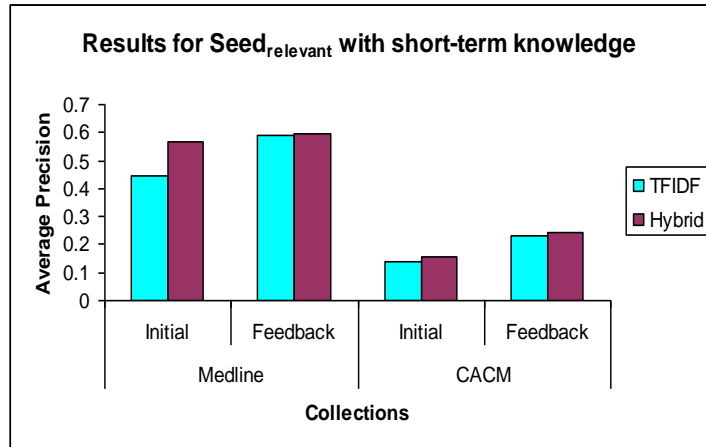
Even within one topic, the queries are about different important problems within that topic. Thus, the values of *averageSim* for those queries selected in the training set in our second method are small. For example, in the MEDLINE collection, the query that relates to highest number of other queries has *averageSim* be 0.0295 while such an *averageSim* is 0.021493 for CACM. This situation implies that even though one query relates to other queries, the relation is very weak.. The number of non-zero similarities queries represents the coverage (how well-connected) of one query to other queries while the *averageSim* represents the overlap (how much in common) that this query actually has with other queries. This also indirectly implies that there is little overlap of the contents of the document sets that are relevant to each query. This prevents $Seed_{relevant}$ from producing better results because (i) some queries do not relate to each other at all and therefore, the knowledge learned from one query can not be reused to find relevant information for another one; and (ii) for some queries that are considered as “related” to each other, their similarities are small. Therefore, only a small portion of knowledge learned is useful in finding relevant information for other queries.

From our experiment, it demonstrates that having more knowledge may not necessarily produce the better retrieval success but having more relevant knowledge to the task at hand is crucial. For example, the biggest context network generated by one individual query included in the training set for $Seed_{knowledge}$ for Medline is 62K while the biggest context network generated by one individual query included in the training for $Seed_{relevant}$ is 32K. The improvement for MEDLINE collection in the initial run, by using $Seed_{knowledge}$ with short-term knowledge is 11.25% and by using $Seed_{relevant}$ with short-term knowledge is 11.86%. Similarly, for CACM collection, the biggest context network generated by one individual query in the training set for $Seed_{knowledge}$ is 40K and the biggest context network generated by a query for $Seed_{relevant}$ is 27K. The improvement for CACM collection in the initial run by using $Seed_{knowledge}$ is 1.7% and by using $Seed_{relevant}$ is 1.9%. In our experiment, we did not find any evidence that supports $Seed_{knowledge}$ over the

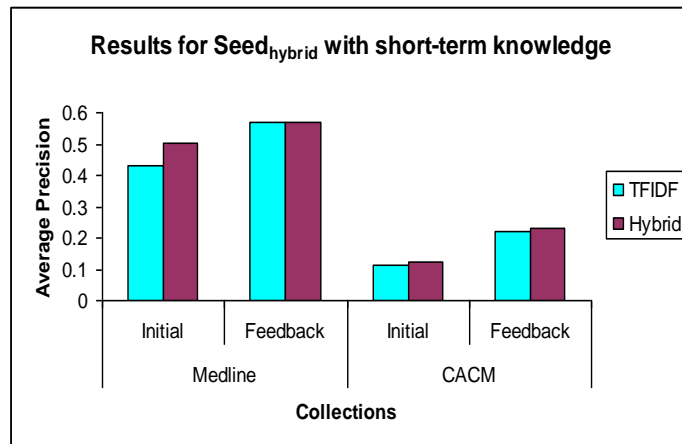
other two seeds with regards to retrieval success. In fact, the $Seed_{relevant}$ performs very competitively when compared to $Seed_{knowledge}$.



(a)

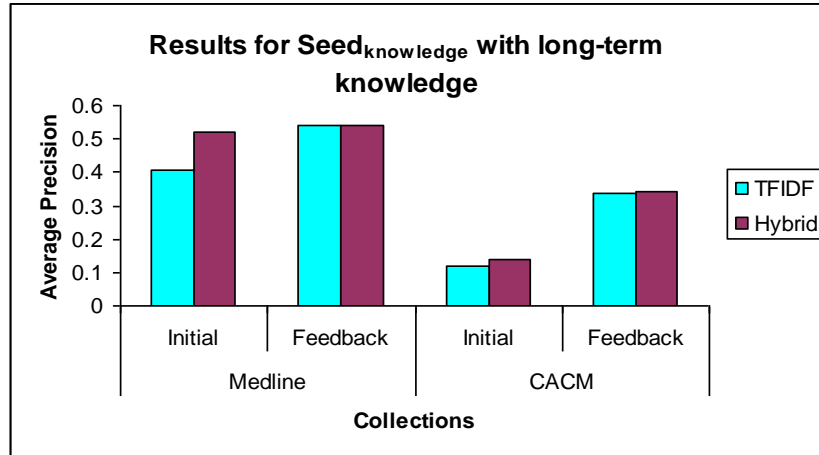


(b)

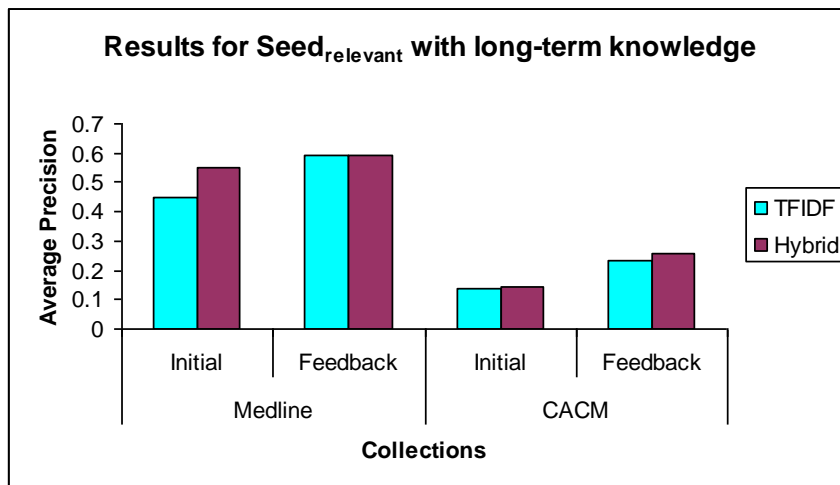


(c)

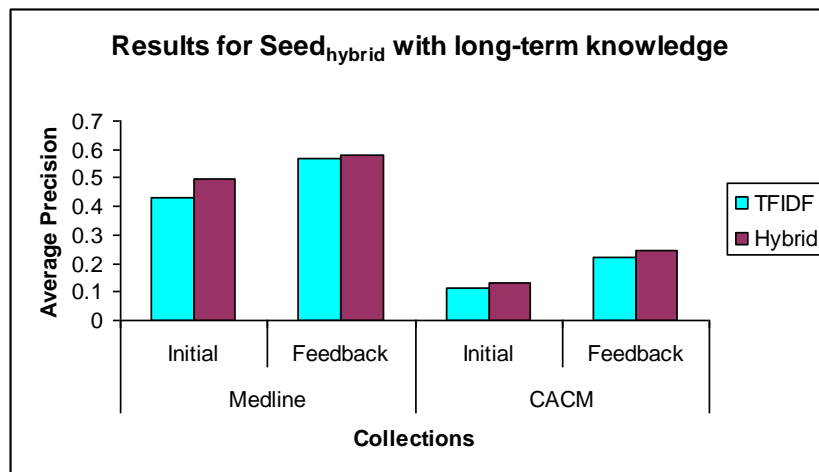
Figure 1: Results for using prior domain knowledge with short-term knowledge.



(a)



(b)



(c)

Figure 2: Results for using prior domain knowledge with long-term knowledge.

6. CONCLUSION and FUTURE WORK

Employing a cognitive user model for IR is a challenging problem to help an IR system to retrieve more relevant documents quickly. Evaluating the effects of a user's domain knowledge that is captured by a user model is a crucial task because it helps us understand what kind of knowledge would be most helpful and most important with regards to improving a user's effectiveness in an information seeking task. In this paper, our findings show that obviously, domain knowledge helps improve the number of retrieved relevant documents. However, capturing more general domain knowledge does not necessarily lead to improvement in a user's effectiveness compared to capturing knowledge that is relevant to the task at hand. This result agrees with other studies that directly evaluate the effects of a real user's domain knowledge on retrieval success²³.

In the future, we would like to extend our study further by constructing a more condensed testbed with highly coherent document collections and a set of queries that are closely related to a small number of topics. We plan to use the CNS¹⁸ collection on Weapons of Mass Destruction also for this evaluation. This collection seems to be a good fit with our user modeling technique because its log session for the search centers around a set of small sub-topic which is in turn associated with smaller topic. Secondly, we want to try different ways of generating Seed_{relevant} and Seed_{knowledge}. Currently, we base our construction on the sizes of the context networks to construct the Seed_{knowledge}. Another way is to base on the number of unique topics that a context network contains. We may also construct Seed_{relevant} differently by choosing the concepts that are either more general or more specific than others in a hierarchy of related concepts. We can use an existing taxonomy or domain ontology for determining this hierarchy.

7. ACKNOWLEDGEMENTS

This work was supported in part by National Geospatial-Intelligence Agency Grant Nos. HM1582-04-1-2027 and HM1582-05-1-2042. Thanks to Nathan Smith and Aaron Schuett for their helpful comments in constructing the training sets.

REFERENCES

- [1] Allen, B.L. Cognitive research in information science: implications for design. *Annual Review of Information Science and Technology*. Vol 26, pp 3-37. 1991.
- [2] Brajnik, G. and Guida, G. and Tasso, C. User modeling in intelligent information retrieval. *Information Processing and Management*. Vol 23(4), pp 305-320. 1987.
- [3] Borgman, C.L. All users of information retrieval systems are not created equal: an exploration into individual differences. *Information Processing & Management*. Vol 25(3), pp 237-251. 1989.
- [4] Boughanem, M and Tmar, M. Incremental adaptive filtering: Profile learning and threshold calibration. *In the Proceedings of SAC 2002*, Madrid, Spain, pp 640-644. 2002.
- [5] Drucker, H. and Shahrany, B. and Gibbon, C. Support vector machines: relevance feedback and information retrieval. *Information Processing and Management*. Vol 38(3), pp 305-323. 2002.
- [6] He, B. and Ounis, I. Inferring Query Performance Using Pre-retrieval Predictors. *In the Proceedings of Information Systems, Special Issue for the String Processing and Information Retrieval: 11th International Conference (SPIRE2004)*. pp 43-54. 2004.
- [7] Hsieh-Yee, I. Effects of Search Experience and Subject Knowledge on Online Search Behavior: Measuring the Search Tactics of Novice and Experienced Searchers. *Journal of the American Society for Information Science*. Vol 44, pp 161-174. 1993.

- [8] Jarvelin K.; Kekalainen, J.; Niemi, T. ExpansionTool: Concept-Based Query Expansion and Construction..*Information Retrieval*. Vol 4(3-4), pp 231-255. 2001.
- [9] Jensen, F. V. An Introduction to Bayesian Networks. University College London Press, London. 1996
- [10] Keeney, L. R.; and Raiffa, H. Decision with Multiple Objectives: Preferences and Value Tradeoffs. John Wiley and Sons. 1976.
- [11] Loper-Pujalte, C. and Guerrero-Bote, V. and DeMoya-Anegon, F. Genetic algorithms in relevance feedback: a second test and new contributions. *Information Processing and Management*. Vol 39(5), pp 669-697. 2003.
- [12] Mitchell, J.F. T.; Chen Y. S.; and Macredie D. R. Hypermedia Learning and prior knowledge: domain expertise vs. system expertise. *Journal of Computer Assisted Learning*. Vol 21, pp 53-64. 2005.
- [13] Nguyen, H., Santos Jr. E., Schuet, A., and Smith, N. Hybrid User Model for Information Retrieval. In *Technical Report of Modeling Others from Observations workshop at Twenty-First National Conference on Artificial Intelligence (AAAI-06) conference*. Boston, MA. 2006.
- [14] Nguyen, H. 2005. Capturing User Intent for Information Retrieval. Ph.D thesis. University of Connecticut.
- [15] Nguyen, H. Santos, Jr., E., Zhao, Q. and Wang, H. Capturing User Intent for Information Retrieval. In *Proceedings of the 48th Annual Meeting for the Human Factors and Ergonomics Society (HFES-04)*. New Orleans, LA, pp 371-375. 2004.
- [16] Salton G. and Buckley C. Improving Retrieval Performance by Relevance Feedback. *Journal of the American Society for Information Science*. Vol 41(4), pp 288-297. 1990.
- [17] Santos Jr., E., Nguyen, H., Zhao, Q. and Pukinskis, E. Empirical Evaluation of Adaptive User Modeling in a Medical Information Retrieval Application. *Lecture Notes in Artificial Intelligence 2702: User Modeling 2003 (Eds. P. Brusilovsky, A. Corbett, and F. de Rosis)*, Springer, pp 292-296. 2003.
- [18] Santos Jr. E., Zhao Q., Nguyen, H., Wang H. Impacts of User Modeling on Personalization of Information Retrieval: An evaluation with human intelligence analysts..In *Technical report of Workshop on Evaluation of Adaptive Systems at User Modeling Conference (UM-2005)*. Edinburgh. Scotland. Pp 27-36. 2005.
- [19] Saracevic, T. and Kantor, P. A study of information seeking and retrieving. III. Searchers, searches, and overlap. *Journal of the American Society for Information Science*. Vol 39(3), pp 197-216. 1988.
- [20] Sleator D. D. and Temperley D. Parsing English with a link Grammar. In *Proceedings of the Third International Workshop on Parsing Technologies*. Pages 277-292. 1993.
- [21] Richardson R. and Smeaton A. Using WordNet in a Knowledge-Based Approach to Information Retrieval. Technical report. School of Computer Applications, Trinity College Dublin. <http://citeseer.ist.psu.edu/richardson95using.html>. 1995.
- [22] Vakkari, P., Pennanen, M., Serola, S. Changes of search terms and tactics while writing a research proposal: a longitudinal case study. *Information Processing and Management*. Vol 39(3), pp 445-463. 2003.
- [23] Wildmuth, B.M. The effects of domain knowledge on search tactic formulation. *Journal of the American Society for Information Science & Technology*. Vol 55(3), pp 246-258. 2004.
- [24] Zhang, X.; Anghelescu, H., Yuan, X. Domain knowledge, search behaviour, and search effectiveness of engineering and science students: an exploratory study. *Information Research*. Volume10(2). Available at: <http://informationr.net/ir/10-2/paper217.html>. 2005

[25] Zhao, Q. Santos Jr., E., Nguyen, H., and Mohammed, A. What Is Needed for a Good Summary?? Two Different Types of Document Sets Yet Seemingly Indistinguishable to Human Users. *In Proceedings of the Thirty-Ninth Annual Hawaii International Conference on Systems Sciences (HICSS-39)*, IEEE Press. Maui, HI. 2006.