Context-Based Topic Models for Query Modification

W. Bruce Croft and Xing Wei

Center for Intelligent Information Retrieval
University of Massachusetts Amherst
140 Governors Drive
Amherst, MA 01002

{croft,xwei}@cs.umass.edu

Abstract

User context, which includes information such as models of user background or interests, is currently a popular research topic in information retrieval (IR), but it is still not clear what its benefits are. This paper demonstrates an empirical upper bound on the potential improvement that context techniques based on modeling the long-term interests of users could achieve. We do this by simulating a user model with a manually selected topic model for each query, and using the topic model to smooth the query. Experiments with these "ideal" topic models on the TREC retrieval tasks show that this type of context model alone provides little benefit, and the overall performance is not as good as relevance modeling (which is a non-context based query modification model). However, smoothing the query with topic models outperforms relevance models for a subset of the queries and automatic selection from these two models for particular queries gives better results overall than relevance models. We further demonstrate some improvements over relevance models with automatically built user models.

Keywords: information retrieval, contextual retrieval, user context, user model, language model

1. Introduction

The aim of contextual retrieval is to "combine search technologies and knowledge about query and user context into a single framework in order to provide the most 'appropriate' answer for a user's information needs" [1]. In a typical retrieval environment, we are given a query and a large collection of documents. The basic IR problem is to retrieve documents relevant to the query. A query is all the information that we have to understand a user's information need and to determine relevance. Typically, a query contains only a few keywords, which are not always good descriptors of content. Given this absence of adequate query information, it is important to consider what other information sources can be exploited to understand the information need, such as context. Contextual retrieval is based on the hypothesis that context information will help describe a user's needs and consequently improve retrieval performance.

There is a variety of context information, such as query features, user background, user interests, etc. This paper focuses on user related information that reflects topical interests, and we refer to this as user context, which is often simply described as "context" or "user profiles" in other papers. The corresponding research field has been called various names such as "personalized IR", "user modeling", "user orientation", "contextual retrieval", etc. In some cases, context is used to refer to short term user interests with respect to specific queries. User profiles, however, can also be used for longer-term, broad topical interests. In this paper, we focus on user models representing longer-term topical interests that can be used to improve specific queries.

User-oriented analytical studies emerged as early as the 1970's [4, 13], but it wasn't until the mid-80's that practical "real world" systems were studied [2]. User oriented approaches and user context information have received more attention recently, including in commercial search engines. For example, Watson [16, 5, 18] predicts user needs and offers relevant information by monitoring the user's actions and capturing content from different applications, such as Internet Explorer and Microsoft Word. The "Stuff I've Seen" system [11] indexes the content seen by a user and provides contextual information for web searches. Google also featured personal history features in its "My Search History" service Beta version [12].

Despite the recent focus on this problem, it is still not clear what the benefits of user context are, especially with collections of realistic size. This paper addresses the question of whether user context can improve IR performance. In order to answer this question, we need to develop user models that have a reasonable potential for improving search. Many methods have been suggested for building user models, based on information such as documents viewed and Web accesses. In order to focus on the *potential* improvement from using context, in our first experiment we chose the "best" topic model for each query

in a set of TREC queries and used this topic model to modify the query using language modeling techniques [7]. The topic model provides background information for the query and, in effect, expands the query with related terms. The use of context information to expand or smooth queries has been used in a number of studies (e.g. [22]). Topic models are based on categories from the Open Directory project [6]. Compared to the type of user models built by observing user behavior, these models should be more focused and less "noisy". We compare these "ideal" context models with the performance of relevance models [17], which are non-user based topic models constructed for each query using the pseudo-relevance feedback approach.

We then examine differences between these two approaches, and whether they can be combined to give better performance. We also examine techniques for automatically selecting a topic model from the Open Directory categories and compare this to the manual selection and relevance model approaches.

Section 2 describes the manually selected context topic models, and presents two algorithms that combine the topic model with the relevance model. The automatically selected topic model and its results are presented in Section 3. Section 4 discusses the results and their implications, and the last section points out possible directions for future work.

2. Effectiveness of User Context: An Empirical Upper Bound

To demonstrate an empirical upper bound, we simulate an "ideal" context model for each query by selecting the "best" topic model for it from the Open Directory project categories [6]. Then we incorporate the model into a language modeling framework as a smoothing or background model for the query. We compare the results with two other techniques in the language modeling framework, which do not use context information, to estimate the potential performance improvement using the context topic models.

In the later part of this section, we examine the combination of the topic model with the relevance model at both model level and query level.

2.1 Constructing Topic Models from the Open Directory

To construct the topic model for each query, we manually select the "closest" categories from the Open Directory project, according to some rules to approximate an "ideal" user model.

2.1.1 Open Directory Project

The Open Directory project [6] (ODP), also known as DMoz (for Directory.Mozilla, the domain name of ODP), is an open content directory of Web links that is constructed and maintained by a community of volunteer editors. It is the largest, most comprehensive human-edited directory of the Web.

An ontology is a specification of concepts and relations between them. ODP uses a hierarchical ontology scheme for organizing site listings. Listings on a similar topic are grouped into categories, which can then include smaller categories. This ontology has been used as the basis of user profiles for personalized search [12, 21].

The Open Directory Project homepage claims that their directory contains more than 500,000 categories, some of which are very specific and small. Trajkova et al. use only the top few levels of the concept hierarchy, and further restrict them to only those concepts that have sufficient data (the web links) associated with them, in their user profile building [21]. In order to build the "best" topic model, we use the whole concept/topic hierachy, but we ignore the categories that contain insufficient data (less than 5 Web links in our experiments). We currently only retrieve the first-level Web pages mentioned in a category without considering further links, to avoid including irrelevant information, and to make the topic model more focused.

2.1.2 Choosing Categories

We want to choose the "closest" categories for a query. "Closest" can be interpreted here as "deepest", that is, there is no applicable category of the query that is deeper (in hierarchy structure) than the currently selected one. In Figure 1, for example, "Energy" is closer than "Technology" to the query of "hydrogen fuel automobiles" (Topic 382 in the TREC7 ad hoc retrieval task) and "Transportation" is closer than "Energy", and there is no sub category in "Transportation" that can cover the query. In this example, "Top/Science/Technology/ Energy/Transportation/" is selected as one of the "closest" categories. For two categories that do not have direct hierarchical relations, their distances to the query are not comparable and both can be selected. For example, both "Transportation" and "Hydrogen" in Figure 1 may be selected.

The above category selection process can be described by two rules:

1) The category should cover the query content.
2) The category should be the closest (deepest in the hierarchical structure) to the query. This provides the most specific/best user information in the Open Directory for this query.

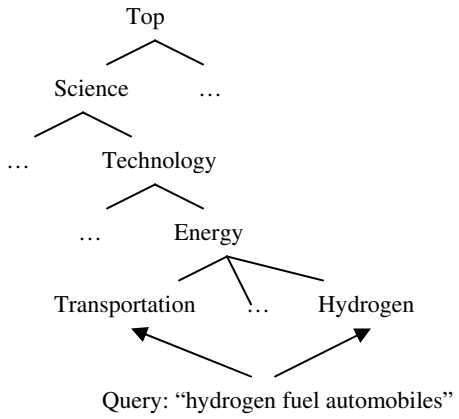

Figure 1: ODP category selection

2.1.3 Constructing Topic Models

After we select the categories for the queries, we download the Web links in the categories we chose. As we said in Section 2.1.1, we download only the first-level pages in the Web links. Then we have a topic collection for each query and we build the topic model U where $P(w|U)$ is estimated by maximum likelihood estimation to be the number of occurrences of w in the topic collection divided by the total number of term occurrences in the topic collection.

To incorporate this topic model, which is a simulated "ideal" user model, into the retrieval framework, a query is smoothed with the topic model to build a modified query. With linear smoothing, we have:

$$P(w \mid Q) = \lambda P'(w \mid Q) + (1-\lambda) P'(w \mid U)$$

where $P'(w|Q)$ is the probability of the word w in the original query model, which is estimated by the number of occurrences of w in query Q divided by the number of total term occurrences in Q. $P'(w|U)$ is the probability of the word in the topic model, which is estimated by the number of occurrences of w in the topic model U. With Dirichlet smoothing [7], we have:

$$P(w \mid Q) = \frac{\|D\|}{\|D\| + \mu} P'(w \mid Q) + (1 - \frac{\|D\|}{\|D\| + \mu}) P'(w \mid U)$$

where $\|D\|$ is the length of the document.

We tried both linear smoothing and Dirichlet smoothing, and chose Dirichlet smoothing with $\mu = 8$ based on empirical evidence. Linear smoothing performs better on some of the experiments but its overall performance is not as consistent as Dirichlet smoothing in our experiments.

After the new query model is built, documents are ranked by the KL divergence between the query model and the document model [7].

In our experiments there are some queries (9 in TREC6, 8 in TREC7 and 15 in TREC8) for which we are unable to find appropriate categories in the Open Directory project, and some queries for which there is insufficient data (too few web links) in the categories we find. We ignore these queries to best estimate the potential performance improvement of user context.

2.2 Baseline Algorithms

We chose two baseline retrieval models: query likelihood and relevance models. Query Likelihood (QL) is a simple retrieval technique and common baseline. Relevance modeling (RM) is an effective query modification technique that fits cleanly into the language modeling framework [7]. We chose relevance modeling as a baseline because it is a non-context based query modification approach. Relevance models modify the queries using the pseudo-feedback approach which relies only on an initial ranking of the documents.

2.2.1 Baseline 1: query likelihood model

We use the query likelihood model where each document is scored by the likelihood of its model generating a query Q.

$$P(Q \mid D) = \prod_{q \in Q} P(q \mid D) \qquad (1)$$

where M_D is a document model, Q is the query and q is a query term in Q. $P(Q|D)$ is the likelihood of a document's model generating the query terms under the assumption that terms are independent given the documents. We construct the document model with Dirichlet smoothing,

$$P(w \mid D) = \frac{\|D\|}{\|D\| + \mu} P'(w \mid D) + (1 - \frac{\|D\|}{\|D\| + \mu}) P'(w \mid coll) \quad (2)$$

where $P'(w|D)$ is the number of occurrences of w in document D divided by the number of total term occurrences in D. $P'(w|coll)$ is the collection probabilities that are estimated using the entire collection. In our experiments, we used a fixed Dirichlet prior with μ=1000.

2.2.2 Baseline 2: relevance model retrieval

The key to relevance model retrieval is estimating the relevance model. Each document is then scored for retrieval by the distance of its model to the relevance model.

Conceptually, the relevance model is a description of an information need or, alternatively, a description of the topic area associated with the information need. From the query modification point of view, the relevance model is the modified query that has a probability (weight) for every term in the vocabulary [17]. It is estimated from the query alone, with no training data, as a weighted average of document models, with the estimates of $P(M_D|Q)$ serving as mixing weights:

$$P(w \mid Q) = \sum_{D \in R} P(w \mid D) P(D \mid Q) \qquad (3)$$

The document models are linearly smoothed (with $\lambda = 0.9$ in this paper),

$$P(w \mid D) = \lambda P'(w \mid D) + (1 - \lambda) P'(w \mid coll) \qquad (4)$$

where $P(D|Q)$ is estimated by Bayes Rule:

$$P(D \mid Q) \propto P(Q \mid D) P(D) \qquad (5)$$

Since P(Q) does not depend on D, the above proportionality holds. With uniform priors, $P(D)$, the posterior probability $P(D|Q)$ amounts to a normalization since we require $P(D|Q)$ to sum to 1 over all documents. $P(Q|D)$ here is from Equation (1).

Then, each document is scored by the cross-entropy of its model to the relevance model. Here the document models over all terms are estimated using linear smoothing with $\lambda = 0.2$ as in Equation (4). All choices of smoothing types and parameters are based on experimental evidence.

Relevance modeling provides a formal method for incorporating query modification into the language model framework, and this approach has achieved good performance in previous experiments [7].

2.3 Experiments

2.3.1 System details

Our experiments were based on TREC ad-hoc retrieval tasks. The data sets include three TREC title query sets: TREC6 (301-350), TREC7 (351-400) and TREC8 (401-450). We indexed the TREC document collections for these data sets using Lemur [19] – a language modeling and information retrieval toolkit. In all experiments, we used the Krovetz [15] stemmer and the default stop word list in Lemur. Retrieval runs are evaluated using trec_eval [22] provided as part of the TREC ad hoc task.

2.3.2 Results

The retrieval performance of manually selected topic models is shown in Table 1 with the baseline results. From the table, we can see that, compared to the query likelihood baseline, the user context topic model shows some improvement for each query set. The improvements for TREC6 and TREC7 are significant based on a t-test. Compared to the relevance model baseline, however, the user context retrieval results are not consistent. On the TREC6 collection, there is some improvement, but results are significantly worse on TREC7 and only the same on TREC8. This demonstrates that even under ideal conditions where the topic model is manually chosen, user context models do not perform better than an automatic method that is user independent. Although this result is limited in that these are not real user models, it certainly casts doubt on the approach of improving queries through context-based background or topic models.

TREC6 queries 301-350 (title)					
	QL	**RM**	**UC**	**%chg (QL)**	**%chg (RM)**
Rel	4611	4611	4611		
Rret	2358	2171	2423	+2.8	+11.6
0.00	0.6768	0.6184	0.7131	+5.4	+15.3
0.10	0.4648	0.4662	0.5	+7.6	+7.3
0.20	0.3683	0.3662	0.3832	+4.1	+4.6
0.30	0.2821	0.2904	0.3305	+17.2	+13.8
0.40	0.2385	0.2495	0.2716	+13.9	+8.9
0.50	0.1906	0.2101	0.2109	+10.7	+0.38
0.60	0.1528	0.1541	0.1693	+10.8	+9.9
0.70	0.1324	0.1088	0.1161	-12	+6.7
0.80	0.0708	0.0597	0.0643	-9.2	+7.7
0.90	0.0423	0.026	0.0412	-2.6	+58.5
1.00	0.0221	0.0108	0.0221	0	+104.6
Avg	0.2193	0.2133	0.2344	+6.99	+9.9
TREC7 queries 351-400 (title)					
	QL	**RM**	**UC**	**%chg (QL)**	**%chg (RM)**
Rel	4674	4674	4674		
Rret	2290	2939	2429	+6.1	-17.4
0.00	0.7221	0.6407	0.7376	+2.2	+15.1
0.10	0.429	0.4861	0.4989	+16.3	+2.6
0.20	0.33	0.3849	0.3613	+9.5	-6.1
0.30	0.2795	0.3316	0.3109	+11.2	-6.2
0.40	0.2177	0.2879	0.2295	+5.4	-20.3
0.50	0.1566	0.2462	0.1681	+7.4	-31.7
0.60	0.1028	0.1949	0.1125	+9.4	-42.3
0.70	0.0683	0.1518	0.081	+8.6	-46.6
0.80	0.0489	0.1099	0.0507	+3.7	-53.9
0.90	0.0384	0.0608	0.0371	-3.4	-39.0
1.00	0.0126	0.0181	0.0131	+4.0	-27.6
Avg	0.1944	0.2515	0.2127	+9.4	-15.4
TREC8 queries 401-450 (title)					
	QL	**RM**	**UC**	**%chg (QL)**	**%chg (RM)**
Rel	4728	4728	4728		
Rret	2764	3085	2835	+2.6	-8.1
0.00	0.7552	0.7097	0.7744	+2.5	+9.1
0.10	0.4979	0.5041	0.5321	+6.9	+5.6
0.20	0.3786	0.411	0.3988	+5.3	-3.0
0.30	0.3235	0.3571	0.3285	+1.6	-8.0
0.40	0.2574	0.304	0.2588	+0.5	-14.9
0.50	0.2246	0.2525	0.2182	-2.8	-13.6
0.60	0.1752	0.191	0.1737	-0.9	-9.1
0.70	0.1397	0.1409	0.1227	-11.5	-12.9
0.80	0.1043	0.0925	0.0983	-5.8	+6.3
0.90	0.0897	0.054	0.0841	-6.2	+55.7
1.00	0.0567	0.0247	0.0465	-18.0	+88.26
Avg	0.2497	0.2546	0.2529	+1.28	-0.67

Table 1. Comparison of the user context (UC) topic model with the query likelihood (QL) model and the relevance model (RM). The evaluation measure is average precision. %chg(QL) denotes the percent change in performance over query likelihood model, and %chg(RM) denotes the change over relevance model.

2.3.3 Result Analysis

A more in-depth analysis of the results gives some indication why the user context model does not perform as well overall as the relevance model. We find that the user context model performs somewhat better on some queries, and much worse on others. Table 2 shows the number of queries that benefit (or suffer) from user context models. Generally, user context models work better on queries that do not have a clear topic, especially those containing words that have several meanings. On the other hand, relevance models work better on queries that are very specific and clear. For example, the query of "mainstreaming" (Topic 379 in the TREC7 ad hoc retrieval task) refers to a special education field, but after stemming this word has multiple meanings not related to education, which results in the system retrieving many irrelevant documents. In this situation, the relevance model technique for modifying the query does not help since there is too much incorrect information. In contrast to this, the manually selected topic model is based on a human interpretation of the query and therefore is focused on the correct meaning.

In the above example, the "ideal" user context model works better. However, there are other queries in which relevance models work better. One such query, "poaching, wildlife preserves" (Topic 407 in the TREC8 ad hoc retrieval task), is very clearly about poaching in wildlife preserves. The initial ranking produces good documents and relevance modeling modifies the query appropriately. User context models also have the potential to work well on these types of queries if there are specific categories in the ODP. In this example, the granularity of the category is much broader than documents. The category closest to this example is "wildlife preserves", which misses the important "poaching" part, and the results are worse than relevance models. Even if we have a specific category related to the query, relevance models can still perform better. The content of the specific category in the Open Directory project can be much less than the relevant documents in the whole collection and the information for query modification that it provides is not as good as the information the collection provides. This is also one of the drawbacks of real user models – usually a user's background is not better than the whole collection, and pseudo-feedback techniques often provide more information than user models.

2.4 Combination

Based on the results and the above analysis, we tried to improve on the relevance model baseline. The user context models are built in an "ideal" simulation, which theoretically, leaves no room for improvement. But from the analysis in Section 2.2.3, we find that the user context model and the relevance model work well on different kinds of queries, which naturally leads to studying some way of combining the advantages of both models. The most straightforward way is to combine these two models at the model level. Another possibility is to employ a technique that selects different models for different queries.

	UC	EQ	RM	Diff
TREC6	28	10	12	+16
TREC7	20	8	22	-2
TREC8	16	15	19	-3

Table 2. Numbers of queries that UC or RM performs better respectively. UC refers to the queries UC performs better and RM refers the ones that RM is better. EQ refers to same performance (most of them are caused by no topic model for the queries). The last column is the difference between column "UC" and column "RM".

2.4.1 Model-level Combination

As described in Section 2.1.3, to compute the relevance models we need $P(Q|D)$ from Equation (1). This is a basic step for relevance model computation. Since we have the user context model, which achieves better performance than the query likelihood model, we replace the query likelihood model with the user context model with query likelihood baseline and complete the other steps as usual. This is a model-level combination, which is denoted by MCOM in Table 3. The average precision is presented in Table 3 and the numbers of queries for which the combination model improves over relevance model are shown in Table 4.

2.4.2 Query-level Combination: Clarity Score Selection

Query modification showing improvement for only some of the queries is a common problem in information retrieval. When examining the results of any query expansion method over a large number of queries, one always finds that nearly equal numbers of queries are helped and hurt by the given technique [10]. Cronen-Townsend et al. developed the clarity metric for choosing which queries benefit most from query expansion techniques [8,9,10]. The weighted clarity score is defined by:

$$clarity = \sum_{w \in V} \frac{u(w)P(w|Q)}{\sum_{w' \in V} u(w')P(w'|Q)} \log_2 \frac{P(w|Q)}{P(w|coll)} \qquad (8)$$

where $u(w)$ are the term weights and V is the vocabulary of the collection. For the algorithm details please refer to [10].

A low clarity score means the query is not very effective and may need modification. In Cronen-Townsend et al.'s original application, the clarity score was used to predict when to use relevance model retrieval to do query modification. According to the analysis in Section 2.2.3, "clear" queries achieve better performance with relevance models and "unclear" queries achieve better performance

with user context models. Thus the clarity score is a reasonable selection method to predict when to use the user context model to do query modification.

TREC6 queries 301-350 (title)				
RM	**MCOM**	**%chg**	**QCOM**	**%chg**
0.2133	0.1817	-14.8	0.2172	+1.8%
TREC7 queries 351-400 (title)				
RM	**MCOM**	**%chg**	**RCOM**	**%chg**
0.2515	0.2596	+3.2%	0.2673	+6.3%
TREC8 queries 401-450 (title)				
RM	**MCOM**	**%chg**	**RCOM**	**%chg**
0.2546	0.2700	+6.0%	0.2573	+1.1%

Table 3. Comparison of the user context model with two combinations. %chg denotes the percent change in performance over RM (measured in average precision).

	MCOM	**EQ**	**RM**	**Diff**
TREC6	19	11	20	-1
TREC7	24	8	18	+6
TREC8	24	14	12	+12
	QCOM	**EQ**	**RM**	**Diff**
TREC6	13	30	7	+6
TREC7	10	35	5	+5
TREC8	9	33	8	+1

Table 4. Numbers of queries that MCOM or RM performs better. MCOM/QCOM refers to the queries MCOM/QCOM performs better and RM refers the ones that RM is better. EQ refers to same performance. The last column is the difference between column "MCOM" and column "RM".

This is a query-level combination, which is represented by QCOM in Table 3. The results of QCOM in Table 3 are plotted in Figure 2 as well. Clarity score selection leads to improvements over relevance models on all three tasks. The improvement is more significant particularly at the top of the ranked list.. This is a good sign since a user often goes through only the documents that are provided first and the documents near to the end plays a less significant role when there are a large number of documents retrieved.

The numbers of queries that are improved (or not improved) by a combination at the query-level, as compared to relevance models, is reported in Table 4. With clarity score selection, more queries benefit from the query-level combination than relevance models on all the three TREC tasks.

3. An Automated Categorization Algorithm

Given that manually selected topic models based on ODP categories showed some promise in our previous results, we also investigated an algorithm for automatically selecting a category for a query. In this case, rather than simulating user context models, we are viewing the ODP categories as an alternative to relevance modeling for automatically smoothing the query (i.e. providing topical context).

3.1 Algorithm

The following is the automated categorization algorithm we used for experiments:

1) Treat the whole open directory as a collection and each category as a document. There are descriptions of the sites in each category, which we treat as the document content (The queries are the original title queries as we used in previous experiments). We retrieved the top 5 categories by query likelihood, and only select the categories from these five.

2) Try to find the categories that are close to the query according to the following rules:

> (a) All the query terms show up in the category name, which is a directory with the category names at each level, e.g., "Top/Computers/Artificial Intelligence/ Applications".
> (b) The most detailed category name, which is "Applications" in the above example, contains only query terms.

3) If we are unable to find the complete categories covering all query terms in the second step, we will use the categories that either have a query likelihood score, computed in 1), larger than a certain threshold, or contain more than half of the query terms.

All the comparisons are made after stemming and stopping. We built topic models as for the hand-selected categories in Section 2, and repeated the experiments on the relevance model baseline with the two combination algorithms.

3.2 Results

The retrieval performance with automated categorization is shown in Table 5 as AC, and the two combination methods are also employed and included for comparison. The numbers of queries that each model works better on are reported in Table 6.

TREC6 queries 301-350 (title)					
RM	**AC**	**MCOM**	**%chg**	**QCOM**	**%chg**
0.2133	0.2299	0.1820	-14.7%	0.2162	+1.4%
TREC7 queries 351-400 (title)					
RM	**AC**	**MCOM**	**%chg**	**QCOM**	**%chg**
0.2515	0.2136	0.2435	-3.2%	0.2534	+0.8%
TREC8 queries 401-450 (title)					
RM	**AC**	**MCOM**	**%chg**	**QCOM**	**%chg**
0.2546	0.2593	0.2661	+4.5%	0.2580	+1.3%

Table 5. Retrieval performance with automated query categorization and two combination algorithms. The evaluation measure is average precision. %chg denotes the percent change in performance over RM.

We found there were slight improvements compared to relevance models. We note that the average precision of AC on TREC8 was better than the manual selection model. Automatic selection of topic models is clearly a viable technique for query smoothing and is complementary to the technique of document smoothing based on cluster models [20] .

An important result is that the clarity score selection again shows good performance again in Table 6, as in Table 4. There are always more queries on which QCOM performs better than relevance models on all the three TREC tasks.

4. Discussion

As described earlier, this paper was motivated by two questions: 1) can user context improve retrieval performance, and 2) how much performance gain can we gain from it. Our experimental results provide some indications of the answers.

	AC	**EQ**	**RM**	**Diff**
TREC6	16	18	16	0
TREC7	20	8	22	-6
TREC8	29	6	15	+14
	MCOM	**EQ**	**RM**	**Diff**
TREC6	14	12	24	+10
TREC7	19	8	23	+4
TREC8	18	19	13	-5
	QCOM	**EQ**	**RM**	**Diff**
TREC6	13	27	10	+3
TREC7	11	32	7	+4
TREC8	10	32	6	+4

Table 6. Numbers of queries that ATUC(RM) or RM performs better, with the comparisons after MCOM and QCOM.

4.1 Can user context improve IR?

In our experiments, the simulated user context model showed some improvement over the query likelihood baseline, but the model itself does not show a consistent or significant improvement over the relevance model baseline. As an "ideal" user context model, the manually selected topic model estimates an empirical upper bound on the benefits of user context modeling when it is used to modify a query. The ideal user models are much more focused than real user models would be. Even given this advantage, this model is inconsistent and is not better overall, compared to relevance modeling, which does not need additional user information. This reflects the difficulty in improving retrieval with user context based on modeling long-term interests.

There is some improvement in the results after combination for the manually selected models, and the advantage of combination was evident even in a simple automatically selected topic model. In Table 4 and Table 6, clarity scores did some useful prediction since the combination approach performs better for the majority of queries.

So, the answer to the first question is that user context in the form of topic models is unlikely to have significant benefits.

4.2 How much gain can we get?

From our experiments, the empirical upper bounds we estimated are not dramatically higher than the relevance model retrieval. Some queries perform well, but many suffer in the user context approaches. In the results after query-level combination, which are relatively consistent, less than 7% improvement is found on average precision, dependent on the TREC tasks. This shows the room for improvement is very limited. The individual upper bound for each query varies a lot. For some queries, the user context model performs very well. The performance improvement of the example query "mainstreaming" we mentioned in Section 2.2.3 is shown in Table 4.

	QL	**RM**	**UC**
Rel	16	16	16
Rret	6	5	14
0.00	0.2	0.0625	1
0.10	0.0292	0.0211	1
0.20	0.0292	0.008	0.5556
0.30	0.0116	0.008	0.5556
0.40	0	0	0.1633
0.50	0	0	0.1633
0.60	0	0	0.0694
0.70	0	0	0.0205
0.80	0	0	0.0186
0.90	0	0	0
1.00	0	0	0
Avg	0.0186	0.0066	0.2756

Table 4. Comparison of performance on query Topic 379

5. Summary

We simulated "ideal" user models to estimate the potential improvement user context could bring to IR in the language model framework. After experimenting with queries from several TREC ad-hoc retrieval tasks, we found that the "ideal" topic models provided little benefit for document retrieval performance compared to relevance models, a non-context based query modification model. In some cases, the user model improves the results, but in other cases relevance models are more effective, and the overall results did not show that user models perform better on these tasks.

Based on the observation that topic models and relevance models benefit different queries, we investigated a combination approach.. Our experiments confirmed that an automatic selection algorithm using the clarity score improves retrieval results.

We also established that topic models based on the ODP categories can be a useful source of information for retrieval. In particular, we showed that smoothing queries

using automatically selected categories improves retrieval performance.

The data and model we used have limitations. Future work will examine other data sets and different situations. Developing new models, especially better combination strategies, is also promising.

6. Acknowledgements

This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF grant #IIS-0527159. The authors would like to thank Vanessa Murdock and Xuerui Wang for helpful discussions and detailed comments. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

7. References

1. Allan, J. et al, "Challenges in Information Retrieval and Language Modeling." Report of a Workshop held at the Center for Intelligent Information Retrieval, University of Massachusetts Amherst, 2002.

2. Belkin, N. and Croft, W.B. "Retrieval techniques." *Annual Review of Information Science and Technology*, Vol.22, 109-145, 1987.

3. Belkin, N., Oddy, R. and Brooks, H. "Ask for Information Retrieval." *Journal of Documentation*, 38, 61-71 (part 1), 145-164 (part 2), 1982.

4. Belkin, N. and Robertson, S.E. "Information Science and the phenomena of Information." *Journal of the American Society for Information Science*, 27, 197-204, 1976.

5. Budzik, J., Hammond K., and Birnbaum, L. "Information access in context." *Knowledge based systems* 14 (1-2), Elsevier Science. 37-53, 2001.

6. http://www.dmoz.com

7. Croft, W.B., and Lafferty, J. editors. *Language Modeling for Information Retrieval*. Kluwer, 11-56, 2003.

8. Cronen-Townsend, S. and Croft, W.B. "Quantifying query ambiguity." *Proceedings. Of the Conference on Human Language Technology HLT 2002*, 94-98, 2002.

9. Cronen-Townsend, S., Zhou, Y., and Croft, W.B. "Predicting query performance." *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, 299-306, 2002.

10. Cronen-Townsend, S., Zhou, Y., and Croft, W.B. "A Language Modeling Framework for Selective Query Expansion." CIIR Technical Report, IR-338, University of Massachusetts, Amherst, MA, 2004.

11. Dumais, S.T., Cutrell, E., Cadiz, J.J., Jancke, G.G., Sarin, R. and Robbins D.C. "Stuff I've Seen: A system for personal information retrieval and re-use." *Proceedings of SIGIR 2003*, 2003.

12. Gauch S., Chaffee J., and Pretschner, A. "Ontology Based Personalized Search." *Web Intelligence and Agent Systems*, 2004.

13. https://www.google.com/searchhistory/login

14. Ingwersen, P. *Information Retrieval Interaction.* London: Taylor Graham, 1992.

15. Ingwersen, P. "Search Procedures in the Library Analysed from the Cognitive Point of View." *Journal of Documentation*, 38, 165-191, 1982.

16. Krovetz, R. "Viewing morphology as an inference process." *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, 191-202, 1993.

17. http://www.intellext.com/

18. Lavrenko, V. and Croft, W.B., "Relevance-based language models." In *Research and Development in Information Retrieval*, W.B. Croft (ed), 120-127, 2001.

19. Leake, D., Scherle, R., Budzik, J., and Hammond, K. J., "Selecting Task-Relevant Sources for Just-in-Time Retrieval." *Proceedings of The AAAI-99 Workshop on Intelligent Information Systems*, AAAI Press, 1999.

20. Liu, X. and Croft, W.B. Liu, X., and Croft, W. B., "Cluster-Based Retrieval Using Language Models." *Proceedings of SIGIR '04*, 186-193, 2004.

21. Ogilvie, P. and Callan, J. "Experiments using the Lemur toolkit." *Proceedings of the 2001 Text Retrieval Conference (TREC 2001)*. 103-108, 2001. http://www.lemurproject.org/

22. Pejtersen, M. "Investigation of Search Strategies in fiction bases on an analysis of 134 user-librarian conversations." *Proceedings of IRFIS 3*, Oslo: Statens Biblioteksskole, 107-131, 1979.

23. Shen, X., and Zhai, C. "Exploiting Query History for Document Ranking in Interactive Information Retrieval." *Proceedings of SIGIR 2003*, 2003.

24. Trajkova, J. and Gauch S. "Improving Ontology-Based User Profiles." *Proceedings of RIAO*, 2004.

25. http://trec.nist.gov/