Investigating Retrieval Performance with Manually-Built Topic Models

Xing Wei & W. Bruce Croft
Center for Intelligent Information Retrieval
University of Massachusetts Amehrst
140 Governors Drive
Amherst, MA 01003
{xwei, croft}@cs.umass.edu

Abstract

Modeling text with topics is currently a popular research area in both Machine Learning and Information Retrieval (IR). Most of this research has focused on automatic methods though there are many hand-crafted topic resources available online. In this paper we investigate retrieval performance with topic models constructed manually based on a hand-crafted directory resource. The original query is smoothed on the manual selected topic model, which can also be viewed as an "ideal" user context model. Experiments with these topic models on the TREC retrieval tasks show that this type of topic model alone provides little benefit, and the overall performance is not as good as relevance modeling (which is an automatic 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 topic models based on the directory resource.

1. INTRODUCTION

Topic models, in which the semantic properties of text are expressed in terms of topics (Steyvers & Griffiths, 2006), have been broadly used in Information Retrieval (IR) systems to improve retrieval performance. A topic is usually represented by a probability distribution over words, where the distribution implies semantic coherence; topics can thus be used as knowledge background which provides semantically related words to expand the literal matching of words that are present in text. The expanded retrieval algorithms can be applied in various IR applications to compensate for literal word-matching algorithms in two ways:

(1) providing general information to address the "vocabulary mismatch" problem. The users of IR systems often use different words to describe the concepts in their queries than the authors use to describe the same or relevant concepts in their documents (Xu, 1997), such as a user may use "apple" as a query and a relevant document may contain "McIntosh" only. A variety of topic models have been built to make up this gap, including using manual/automatic thesauri, dimensionality reduction, (pseudo-)relevance feedback, and latent mixture modeling techniques, through query expansion and/or document expansion methods (Sparck Jones, 1971; Qiu & Frei, 1993; Xu & Croft, 1996; Jing & Croft, 1994; Cao et al., 2005; Deerwester et al., 1990; Hoffman, 1999; Lavrenko, 2001; Wei & Croft 2006).

(2) providing user specific information to integrate user context. The goal of Information Retrieval is to retrieve documents relevant to a user's information need. User context, which refers to user related information that reflects topical interests, is an important information source in addition to queries to help in understanding a user's information need and to determine relevance. The query "apple" that was input by a user who has a computer science background may be different from the query "apple" that was input by a user who has a food science background, and topical context can help differentiating these two queries. User context information has received considerable attention recently, especially in commercial

search engines, such as Watson[1] (Budzik et al., 2001; Leake et al., 1999), Google[2], and "Stuff I've Seen" (Dumais et al., 2003).

In summary, topic models can help the retrieval process by providing additional information to present words, which can be either general knowledge like word meaning/common sense (as the connection between "apple" and "McIntosh") or user oriented information. Although a number of studies have been conducted on these two aspects for topic models, the effectiveness of topic models is still not clear, especially on collections of realistic size. Most recent research on topic models has focused on automatic techniques. To give a broader picture of the potential effectiveness of these approaches, in this paper we investigate the use of manually-built topic models. In real-world IR applications building topic models by hand is often infeasible due to its prohibitive price. Even the simplistic manual topic representation – hand-crafted thesauri are limited by the construction and maintainence price. However, through the popularization of Internet in recent years, topicalized information like the directory service offered by many web sites, has become a significant information resource with reasonable quality, which makes it easier to build topic models manually and also makes it interesting to see how much improvements we can get from this information. Also, manual processing is flexible and capable of generating appropriate topic models including both of general knowledge and user context. So the results can benefit both research directions:

From the point of view of general topic models, the success of automatically-built topic models (usually built on the experimental collection) makes it interesting to see the performance gain with manual methods. Some semi-manual methods have been applied in previous research based on hand-crafted thesauri (e.g., Kwon et al.,1994; cao et al., 2005), which can be viewed as a simplistic topic representation. In this paper we will use manually-constructed directory service, which is a more generous topic representation compared to thesauri, and assign topics/directories to text by hand. So the process can show the effectiveness of fully manual methods, which reflects the potential improvement from using available hand-crafted topic resource since manual processing can build "ideal" topic models from the available resource.

From the point of view of user specific topic models, the manually-built topic models can be viewed as "ideal" context models. Compared to the type of user models built by observing user behavior, these models should be more focused and less "noisy". Also, considering that the available resource may contain insufficient information for some topics, in our experiments we discard the queries for which the resource does not contain sufficient data in order to generate "ideal" context models and thus produce an empirical upper bound for retrieval performance gain with user modeling.

In other words, we focus on the potential improvement from using some well-organized and pre-available resource to form topic models for retrieval. In our first experiment we choose the "best" topic model for each query in a set of TREC queries and use this topic model to modify the query using language modeling techniques (Ponte & Croft, 1998). The topic model provides background information for the query and, in effect, expands the query with related terms. The use of general topic models or context information to expand queries has been used in a number of studies (e.g., Bai et al., 2004; Shen & Zhai, 2003). Topic models are based on categories from the Open Directory project[3] (ODP). We compare these "ideal" topic models with the

[1] http://www.intellext.com/
[2] http://www.google.com/psearch
[3] http://www.dmoz.com/

performance of relevance models (RMs), which are non-user based topic models constructed automatically 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.

Related work is discussed in Section 2 from the perspective of general topic models and user context respectively. Section 3 describes the manually selected topic models, and presents two algorithms that combine the topic model with the relevance model. Automatically selected topic model results are presented in Section 4. Section 5 discusses the results and their implications, and the last section points out possible directions for future work.

2. RELATED WORK

2.1 General Topic Models in IR

The earliest method of incorporating general topic models in IR was by using terms from hand-crafted thesauri. Manual indexing has often been viewed as a gold standard and a thesaurus as a "correct" way of incorporating new words or phrases, but building a thesaurus is very labor-intensive and it is very difficult to get people to agree on the semantic classifications involved. Inconsistencies and ambiguity in the use of these thesauri have produced poor results when they are used for retrieval experiments. In a manually built thesaurus, two words are usually regarded as related when their meanings have something in common; however, two words are actually also related if they are brought together in the context of a topic or subject of discourse: they are related by their shared reference (Jones, 1971). Therefore, a directory service which is based on documents may provide a better profile of the general connections among words, and it can also provide the user context information that thesauri do not have.

Given the difficulties of constructing thesauri manually, people hoped to obtain topic models more easily and effectively by automatic data-driven techniques. Many word similarity measures have been developed, including vector-based similarity coefficients (Qiu and Frei, 1993; Jones, 1971), linguistic-based analysis such as using head-modifier relationships to determine similarity (Grefenstette, 1992), and probabilistic co-occurrence models (Rijsbergen, 1977; Cao et al., 2005). They can be used to find "close" terms based on their metrics and group the terms into clusters/topics, or build a word distribution for each term based on the similarity between the term and other words. Then topic models can be easily generated (e.g., by replacing each term with all the words occurring in the cluster/topic to which it belongs) to conduct IR tasks. Quite a few interesting retrieval results have been achieved with this type of techniques.

Grouping terms is a straightforward approach to finding related words for topic models; grouping documents, at the same time, has also been effectively used to build topic models by either constructing term clusters based on document clusters (Crouch, 1990) or viewing a document cluster as a topic, and then all documents in the cluster having the identical topic model (Croft, 1980; Liu and Croft, 2004). Liu and Croft (2004) demonstrated that document clustering can achieve significant and consistent improvements over document-based retrieval models in the language modeling framework.

Latent Semantic Analysis or LSA (Deerwester et al., 1990) is an approach that combines both term- and document clustering. LSA usually takes a term-document matrix in the vector space representation (Salton and Mcgill, 1983) as input, and applies singular value decomposition (SVD)-based dimensionality reduction techniques to the matrix. Thus documents and terms are mapped to a representation in the latent semantic space, which is based on topics rather than individual terms and thus much smaller than the original representation space. As an important and novel topic modeling technique, LSA has been heavily cited in many areas including IR and inspired many new research directions. It has been applied into a range of applications and interesting retrieval results on small collections have been achieved with automatic indexing with LSA (Latent Semantic Indexing, LSI, Deerwester et al., 1990; Dumais et al., 1995).

The probabilistic Latent Semantic Indexing (pLSI) model introduced by Hoffman (1999) was designed as a discrete counterpart of LSI to provide a better fit to text data. pLSI is a latent variable model that models each document as a mixture of topics. Another latent topic model, Latent Dirichlet Allocation (LDA, Blei et al., 2003), which overcomes some problems with the generative semantics of pLSI, has quickly become one of the most popular text modeling techniques. Both of pLSI and LDA have shown significant performance on IR tasks (Hoffman, 1999; Wei & Croft, 2006).

The effectiveness of automatic topic models in IR, especially the ones which are not only based on term-term relationships (e.g. document clustering and LDA), makes is very interesting to investigate the retrieval performance with manually-built topic models other than hand-crafted thesauri.

2.2 Contextual Retrieval

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" (Allan, et al., 2002). 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 (Belkin & Robertson, 1976; Pejtersen, 1979; Ingwersen, 1992), but it wasn't until the mid-80's that practical "real world" systems were studied (Belkin & Croft, 1987). User orientated approaches and user context information has received more attention recently, including in commercial search engines. For

example, Watson[4] (Budzik et al., 2001; Leake et al., 1999) 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 (Dumais et al., 2003) indexes the content seen by a user and provides contextual information for web searches. Google[5] also featured personal history features in its "My Search History" service Beta version.

Despite the recent focus on this problem, it is still not clear what the benefits of user context are, especially with test collections of realistic size.

3. EFFECTIVENESS OF MANUALLY-BUILT TOPIC MODELS

To show the potential improvements of the available topic resource and demonstrate an empirical upper bound of using user context in IR, we simulate an "ideal" topic model for each query by selecting the "best" topics for it from the Open Directory project categories. 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 other resources or context information, to estimate the potential performance improvement using.

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.

3.1 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.

3.1.1 Open Directory Project

The Open Directory project (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 (Trajkova & Gauch, 2004).

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 (Trajkova & Gauch, 2004). In order to build the "best" topic model, we use the whole concept/topic hierachy, but we ignore the categories that contains insufficent 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.

[4] http://www.intellext.com/
[5] http://www.google.com/psearch

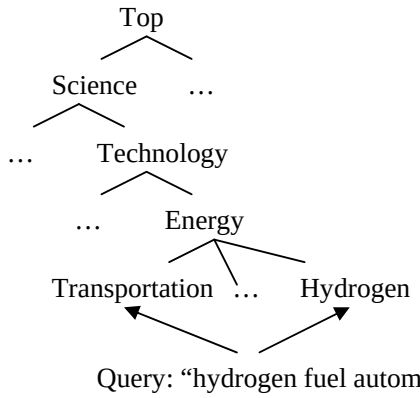

Figure 1. An example of hierarchical categories.

3.1.2 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 information in the Open Directory for this query.

3.1.3 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 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 (Zhai & Lafferty, 2001), 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 μ=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 (Croft & Lafferty, 2003).

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.

3.2 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 (Croft & Lafferty, 2003). 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.

3.2.1 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,

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

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.

3.2.2 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 (Lavrenko, 2001). It is estimated from the query alone, with no training data, as a weighted average of document models, with the estimates of $P(D \mid Q)$ 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 λ=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 KL-divergence of its model to the relevance model. Here the document models over all terms are estimated using linear smoothing with λ=0.9 as in Equation (4). All the choices of soothing 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 (Lavrenko, 2001).

3.3 Experiments

3.3.1 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 (Ogilvie & Callan, 2001) – a language modeling and information retrieval toolkit. In all experiments, we used the Krovetz (Krovetz, 1993) stemmer and the default stop word list in Lemur. Retrieval runs are evaluated using trec_eval[6] provided as part of the TREC ad hoc task.

[6] http://trec.nist.gov/

3.3.2 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 manually-built topic model shows some improvement for each query set. Compared to the relevance model baseline, however, the retrieval results with manually-built topic models 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, topic models based on the directory service do not perform better than an automatic method that is user independent. Although this result is limited in that the directory service could be improved or these are not real user models, it certainly casts doubt on the approach of improving queries through pre-defined topic resources or context-based background.

3.3.3 Result Analysis

A more in-depth analysis of the results gives some indication why the manually-built topic model does not perform as well overall as the relevance model. We find that the manually-built topic model performs somewhat better on some queries, and much worse on others. Table 2 shows the number of queries that benefit (or suffer) from manuall-built topic models. Generally, manually-built topic 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" topic 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. Manually-built topic 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.

Table 1. Comparison of retrieval with the manually-built topic model(MT) 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 QL, and %chg(RM) denotes the change over RM.

TREC6 queries 301-350 (title)					
	QL	**RM**	**MT**	**%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**	**MC**	**%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 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. The last column is the difference between column "UC" and column "RM".

	UC	**EQ**	**RM**	**Diff**
TREC6	32	1	17	+15
TREC7	25	0	25	0
TREC8	22	0	28	-6

3.4 Combination

Based on the results and the above analysis, we tried to improve on the relevance model baseline. The manually-built topic 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 manually-built topic 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.

3.4.1 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 manually-built topic model, which achieves better performance than the query likelihood model, we replace the query likelihood model with the manually-built topic model retrieval 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.

3.4.2 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 (Cronen-Townsend et al., 2004). Cronen-Townsend et al. developed the clarity metric for choosing which queries benefit most from query expansion techniques (Cronen-Townsend & Croft, 2002 ; Cronen-Townsend et al., 2002; Cronen-Townsend et al., 2004). 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 (Cronen-Townsend et al., 2004).

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 manually-built topic models. Thus the clarity score is a reasonable selection method to predict when to use the topic model to do query modification.

This is a query-level combination, which is represented by QCOM in Table 3.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.

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

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**	**QCOM**	**%chg**
0.2515	0.2596	+3.2%	0.2673	+6.3%
TREC8 queries 401-450 (title)				
RM	**MCOM**	**%chg**	**QCOM**	**%chg**
0.2546	0.2700	+6.0%	0.2573	+1.1%

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".

	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

4. AN AUTOMATED CATEGORIZATION ALGORITHM

Given that manually-built 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 "ideal" topic models or 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).

4.1 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.

4.2 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.

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.

TREC6 queries 301-350 (title)					
RM	**AC**	**MCOM**	**%chg**	**QCOM**	**%chg**
0.2133	0.2267	0.1820	-14.7%	0.2162	+1.4%
TREC7 queries 351-400 (title)					
RM	**AC**	**MCOM**	**%chg**	**QCOM**	**%chg**
0.2515	0.1959	0.2435	-3.2%	0.2534	+0.8%
TREC8 queries 401-450 (title)					
RM	**AC**	**MCOM**	**%chg**	**QCOM**	**%chg**
0.2546	0.2545	0.2661	+4.5%	0.2580	+1.3%

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 (Liu & Croft, 2004).

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.

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

	AC	**EQ**	**RM**	**Diff**
TREC6	31	0	19	+12
TREC7	23	0	27	-4
TREC8	22	0	28	-4
	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

5. DISCUSSION

As described earlier, this paper aims at the effectiveness of manually-built topic models, which can be viewed as an "ideal" usage of available topic resources and also an "ideal" user context model. So we are interested in the following two questions: 1) can these topic models, which represent hand-crafted topic resource and user context, improve retrieval performance, and 2) how much performance gain can we gain from them. Our experimental results provide some indications of the answers.

5.1 Can topic resource/user context improve IR?

In our experiments, the manually-built topic 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" manual topic/user context model, the topic models estimates an empirical upper bound on the benefits of hand-crafted topic resrouce/user context modeling when it is used to modify a query. Besides, 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.

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 topic resrouce/user context in the form of topic models is unlikely to have significant benefits.

5.2 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 manually-built topic model performs very well. The performance improvement of the example query “mainstreaming” we mentioned in Section 2.2.3 is shown in Table 7 and plotted in Figure 2.

Table 7. Comparison of performance on query Topic 379

	QL	RM	MT
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

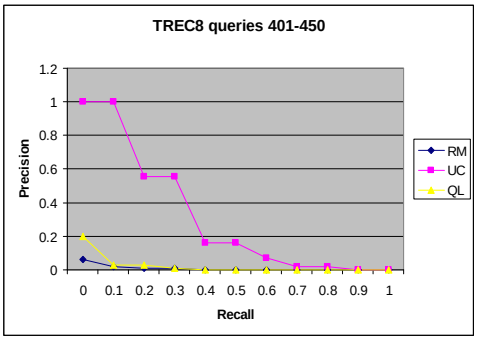

Figure 2. QL, RM and MT performance on Topic 379

6. SUMMARY

We built topic models manually based on a topic resource, which is also hand-crafted, to estimate the potential improvements those hand-crafted topic resources could bring to IR in the language modeling framework, and the result also reflects the potential improvement of user context by viewing the topic models as simulated “ideal” user context models. After experimenting with queries from several TREC ad-hoc retrieval tasks, we found that the manually-built topic models provided little benefit for document retrieval performance compared to relevance models, an automatic non-extra resource based query modification model. In some cases, the topic model improves the results, but in other cases relevance models are more effective, and the overall results did not show that manually-built topic models perform better on these tasks.

Based on the observation that manually-built 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.

7. ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF grant #CNS-0454018 . Any opinions, findings and conclusions or recommendations expressed in this material are the author's and do not necessarily reflect those of the sponsor.

8. REFERENCES

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

Bai, J., Song, D., Bruza, P., Nie, J., and Cao, G. (2005). Query Expansion Using Term Relationships in Language Models for Information Retrieval, In Proceedings of ACM-CIKM 2005, 688-695.

Belkin, N. & Croft, W.B (1987). Retrieval techniques ARIST, Vol.22, p.109-145.

Belkin, N. & Robertson, S.E. (1976). Information Science and the phenomena of Information. *JASIS*, 197-204.

Blei, D. M., Ng, A. Y., and Jordan, M. J. (2003). Latent Dirichlet allocation. In Journal of Machine Learning Research, 3, 993-1022.

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

Cao, G., Nie, J., and Bai, J. (2005). Integrating word relationships into language models. In Proceedings of ACM SIGIR 2--5, 298-305.

Croft, W. B. (1980). A model of cluster searching based on classification. Information Systems, Vol. 5, 189-195.

Croft, W.B., and Lafferty, J. (2003). *Language Modeling for Information Retrieval*. Kluwer, p. 11-56.

Cronen-Townsend, S. and Croft, W.B. (2002). Quantifying query ambiguity. In *Proceedings of Human Language Technology 2002*, 94-98.

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

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

Crouch, C. J. (1990). An approach to the automatic construction of global thesauri. Information Processing & Management, 26(5), 629-640.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R., (1990). Indexing by latent semantic analysis. Journal of the American Society for Information Science, 41(6), 391-407.

Dumais, S. T. (1995). Latent semantic indexing (lsi): Trec-3 report. In Proceedings of the Text Retrieval Conference (TREC-3). 219-230.

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

Gauch S., Chaffee J., and Pretschner A. (2004). Ontology-Based Personalized Search and Browsing. *Web Intelligence and Agent Systems*, Vol. 1 No. 3-4, 219-234.

Grefenstette, G. (1992). Use of syntactic context to produce term association lists for text retrieval. In *Proceedings of SIGIR '92*, 89-97.

Hofmann, T. (1999). Probabilistic latent semantic indexing. In Proceedings of SIGIR '99, Berkeley, CA, USA.

Ingwersen, P. (1992). *Information Retrieval Interaction*. London: Taylor Graham. 1992.X, 246p.

Jones, K. S., Automatic Keyword Classification for Information Retrieval. London: Butterworths, 1971.
Krovetz, R. (1993). Viewing morphology as an inference process. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, 191-202.
Kwon, O.W., Kim, M.C., and Choi Key Sun (1994) Query expansion using domain-adapted, weighted thesaurus in an extended Boolean model. In *Proceedings of CIKM'94*, 140-146.
Lavrenko, V. and Croft, W.B. (2001). Relevance-based language models. In *Research and Development in Information Retrieval*, 120-127.
Leake, D., Scherle, R., Budzik, J., and Hammond, K. J. (1999). Selecting Task-Relevant Sources for Just-in-Time Retrieval. In *Proceedings of The AAAI-99 Workshop on Intelligent Information Systems*, AAAI Press.
Liu, X., and Croft, W. B. (2004). Cluster-Based Retrieval Using Language Models, In *Proceedings of SIGIR '04*, 186-193.
Ogilvie, P. and Callan, J. (2001). Experiments using the Lemur toolkit. In *Proceedings of the 2001 Text Retrieval Conference (TREC 2001)*. 103-108. http://www.lemurproject.org/
Pejtersen, M. (1979). Investigation of Search Strategies in fiction bases on an analysis of 134 user-librarian conversations. In IRFIS 3 Proc. Oslo: Statens Biblioteksskole, 107-131.
Qui, Y. and Frei, H. (1993). Concept based query expansion, In *Proceedings of ACM SIGIR 1993*, 160-169.
Salton, G. and Lesk, M. (1971). *Computer evaluation of indexing and text processing*. Prentice-Hall, 143-180.
Salton, G., and Mcgill, M. J. (1983). Introduction to Modern Information Retrieval. McGraw-Hill.
Shen, X., and Zhai, C. (2003). Exploiting Query History for Document Ranking in Interactive Information Retrieval. In *Proceedings of SIGIR 2003*.
Trajkova, J. and Gauch S. (2004). Improving Ontology-Based User Profiles. *RIAO 2004*, Vaucluse, France, 380-389.
van RIJSBERGEN, C.J. (1977). A theoretical basis for the use of co-occurrence data in information retrieval, In Journal of Documentation, 33, 106-119.
Voorhees, E. M. (1994). Query expansion using lexical-semantic relations. In *Proceedings of SIGIR'94*, 61-69.
Zhai, C. and Lafferty, J. (2001). A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of ACM SIGIR 2001*, 334-342.