# Term Feedback for Information Retrieval with Language Models

Bin Tan[†], Atulya Velivelli[‡], Hui Fang[†], ChengXiang Zhai[†]
Dept. of Computer Science[†], Dept. of Electrical and Computer Engineering[‡]
University of Illinois at Urbana-Champaign
bintan@cs.uiuc.edu, velivell@ifp.uiuc.edu, hfang@cs.uiuc.edu, czhai@cs.uiuc.edu

## ABSTRACT

In this paper we study term-based feedback for information retrieval in the language modeling approach. With term feedback a user *directly* judges the relevance of individual terms without interaction with feedback documents, taking full control of the query expansion process. We propose a cluster-based method for selecting terms to present to the user for judgment, as well as effective algorithms for constructing refined query language models from user term feedback. Our algorithms are shown to bring significant improvement in retrieval accuracy over a non-feedback baseline, and achieve comparable performance to relevance feedback. They are helpful even when there are no relevant documents in the top.

## Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Retrieval models

## General Terms

Algorithms

## Keywords

Query expansion, interactive retrieval

## 1. INTRODUCTION

In the language modeling approach to information retrieval, feedback is often modeled as estimating an improved query model or relevance model based on a set of feedback documents [25, 13]. This is in line with the traditional way of doing relevance feedback - presenting a user with documents/passages for relevance judgment and then extracting terms from the judged documents or passages to expand the initial query. It is an *indirect* way of seeking user's assistance for query model construction, in the sense that the refined query model (based on terms) is learned through feedback documents/passages, which are high-level structures of terms. It has the disadvantage that irrelevant terms, which occur along with relevant ones in the judged content, may be erroneously used for query expansion, causing undesired effects. For example, for the TREC query "Hubble telescope achievements", when a relevant document talks more about the telescope's repair than its discoveries, irrelevant terms such as "spacewalk" can be added into the modified query.

We can consider a more direct way to involve a user in query model improvement, without an intermediary step of document feedback that can introduce noise. The idea is to present a (reasonable) number of individual terms to the user and ask him/her to judge the relevance of each term or directly specify their probabilities in the query model. This strategy has been discussed in [15], but to our knowledge, it has not been seriously studied in existing language modeling literature. Compared to traditional relevance feedback, this term-based approach to interactive query model refinement has several advantages. First, the user has better control of the final query model through direct manipulation of terms: he/she can dictate which terms are relevant, irrelevant, and possibly, to what degree. This avoids the risk of bringing unwanted terms into the query model, although sometimes the user introduces low-quality terms. Second, because a term takes less time to judge than a document's full text or summary, and as few as around 20 presented terms can bring significant improvement in retrieval performance (as we will show later), term feedback makes it faster to gather user feedback. This is especially helpful for interactive ad-hoc search. Third, sometimes there are no relevant documents in the top $N$ of the initially retrieved results if the topic is hard. This is often true when $N$ is constrained to be small, which arises from the fact that the user is unwilling to judge too many documents. In this case, relevance feedback is useless, as no relevant document can be leveraged on, but term feedback is still often helpful, by allowing relevant terms to be picked from irrelevant documents.

During our participation in the TREC 2005 HARD Track and continued study afterward, we explored how to exploit term feedback from the user to construct improved query models for information retrieval in the language modeling approach. We identified two key subtasks of term-based feedback, i.e., pre-feedback presentation term selection and post-feedback query model construction, with effective algorithms developed for both. We imposed a secondary cluster structure on terms and found that a cluster view sheds additional insight into the user's information need, and provides a good way of utilizing term feedback. Through experiments we found that term feedback improves significantly over the non-feedback baseline, even though the user often makes mistakes in relevance judgment. Among our algorithms, the one with best retrieval performance is TCFB, the combination of TFB, the direct term feedback algorithm, and CFB, the cluster-based feedback algorithm. We also varied the number of feedback terms and observed reasonable improvement even at low numbers. Finally, by comparing term feedback with document-level feedback, we found


SIGIR'07, July 23–27, 2007, Amsterdam, The Netherlands.

it to be a viable alternative to the latter with competitive retrieval performance.

The rest of the paper is organized as follows. Section 2 discusses some related work. Section 4 outlines our general approach to term feedback. We present our method for presentation term selection in Section 3 and algorithms for query model construction in Section 5. The experiment results are given in Section 6. Section 7 concludes this paper.

## 2. RELATED WORK

Relevance feedback[17, 19] has long been recognized as an effective method for improving retrieval performance. Normally, the top $N$ documents retrieved using the original query are presented to the user for judgment, after which terms are extracted from the judged relevant documents, weighted by their potential of attracting more relevant documents, and added into the query model. The expanded query usually represents the user's information need better than the original one, which is often just a short keyword query. A second iteration of retrieval using this modified query usually produces significant increase in retrieval accuracy. In cases where true relevance judgment is unavailable and all top $N$ documents are assumed to be relevant, it is called *blind* or *pseudo feedback*[5, 16] and usually still brings performance improvement.

Because document is a large text unit, when it is used for relevance feedback many irrelevant terms can be introduced into the feedback process. To overcome this, *passage feedback* is proposed and shown to improve feedback performance[1, 23]. A more direct solution is to ask the user for their relevance judgment of feedback terms. For example, in some relevance feedback systems such as [12], there is an interaction step that allows the user to add or remove expansion terms after they are automatically extracted from relevant documents. This is categorized as *interactive query expansion*, where the original query is augmented with user-provided terms, which can come from direct user input (free-form text or keywords)[22, 7, 10] or user selection of system-suggested terms (using thesauri[6, 22] or extracted from feedback documents[6, 22, 12, 4, 7]).

In many cases term relevance feedback has been found to effectively improve retrieval performance[6, 22, 12, 4, 10]. For example, the study in [12] shows that the user prefers to have explicit knowledge and direct control of which terms are used for query expansion, and the *penetrable* interface that provides this freedom is shown to perform better than other interfaces. However, in some other cases there is no significant benefit[3, 14], even if the user likes interacting with expansion terms. In a simulated study carried out in [18], the author compares the retrieval performance of interactive query expansion and automatic query expansion with a simulated study, and suggests that the potential benefits of the former can be hard to achieve. The user is found to be not good at identifying useful terms for query expansion, when a simple term presentation interface is unable to provide sufficient semantic context of the feedback terms.

Our work differs from the previous ones in two important aspects. First, when we choose terms to present to the user for relevance judgment, we not only consider single-term value (e.g., the relative frequency of a term in the top documents, which can be measured by metrics such as Robertson Selection Value and Simplified Kullback-Leibler Distance as listed in [24]), but also examine the cluster structure of the terms, so as to produce a balanced coverage of the different topic aspects. Second, with the language modelling framework, we allow an elaborate construction of the updated query model, by setting different probabilities for different terms based on whether it is a query term, its significance in the top documents, and its cluster membership. Although techniques for adjusting query term weights exist for vector space models[17] and probablistic relevance models[9], most of the aforementioned works do not use them, choosing to just append feedback terms to the original query (thus using equal weights for them), which can lead to poorer retrieval performance. The combination of the two aspects allows our method to perform much better than the baseline.

The usual way for feedback term presentation is just to display the terms in a list. There have been some works on alternative user interfaces. [8] arranges terms in a hierarchy, and [11] compares three different interfaces, including terms + checkboxes, terms + context (sentences) + checkboxes, sentences + input text box. In both studies, however, there is no significant performance difference. In our work we adopt the simplest approach of terms + checkboxes. We focus on term presentation and query model construction from feedback terms, and believe using contexts to improve feedback term quality should be orthogonal to our method.

## 3. GENERAL APPROACH

We follow the language modeling approach, and base our method on the KL-divergence retrieval model proposed in [25]. With this model, the retrieval task involves estimating a query language model $\theta_q$ from a given query, a document language model $\theta_d$ from each document, and calculating their KL-divergence $D(\theta_q||\theta_d)$, which is then used to score the documents. [25] treats relevance feedback as a query model re-estimation problem, i.e., computing an updated query model $\theta_{q'}$ given the original query text and the extra evidence carried by the judged relevant documents. We adopt this view, and cast our task as updating the query model from user term feedback. There are two key subtasks here: First, how to choose the best terms to present to the user for judgment, in order to gather maximal evidence about the user's information need. Second, how to compute an updated query model based on this term feedback evidence, so that it captures the user's information need and translates into good retrieval performance.

## 4. PRESENTATION TERM SELECTION

Proper selection of terms to be presented to the user for judgment is crucial to the success of term feedback. If the terms are poorly chosen and there are few relevant ones, the user will have a hard time looking for useful terms to help clarify his/her information need. If the relevant terms are plentiful, but all concentrate on a single aspect of the query topic, then we will only be able to get feedback on that aspect and missing others, resulting in a breadth loss in retrieved results. Therefore, it is important to carefully select presentation terms to maximize expected gain from user feedback, i.e., those that can potentially reveal most evidence of the user's information need. This is similar to *active feedback*[21], which suggests that a retrieval system should actively probe the user's information need, and in the case of relevance feedback, the feedback documents should be chosen to maximize learning benefits (e.g. diversely so as to increase coverage).

In our approach, the top $N$ documents from an initial retrieval using the original query form the source of feedback terms: all terms that appear in them are considered candidates to present to the user. These documents serve as pseudo-feedback, since they provide a much richer context than the original query (usually very short), while the user is not asked to judge their relevance. Due to the latter reason, it is possible to make $N$ quite large (e.g., in our experiments we set $N = 60$) to increase its coverage of different aspects in the topic.

The simplest way of selecting feedback terms is to choose the most frequent $M$ terms from the $N$ documents. This method, however, has two drawbacks. First, a lot of common noisy terms will be selected due to their high frequencies in the document collection, unless a stop-word list is used for filtering. Second, the presentation list will tend to be filled by terms from major aspects of the topic; those from a minor aspect are likely to be missed due to their relatively low frequencies.

We solve the above problems by two corresponding measures. First, we introduce a background model $\theta_B$ that is estimated from collection statistics and explains the common terms, so that they are much less likely to appear in the presentation list. Second, the terms are selected from multiple clusters in the pseudo-feedback documents, to ensure sufficient representation of different aspects of the topic.

We rely on the mixture multinomial model, which is used for theme discovery in [26]. Specifically, we assume the $N$ documents contain $K$ clusters $\{C_i | i = 1, 2, \cdots K\}$, each characterized by a multinomial word distribution (also known as unigram language model) $\theta_i$ and corresponding to an aspect of the topic. The documents are regarded as sampled from a mixture of $K + 1$ components, including the $K$ clusters and the background model:

$$p(w|d) = \lambda_B p(w|\theta_B) + (1 - \lambda_B) \sum_{i=1}^{K} \pi_{d,i} p(w|\theta_i)$$

where $w$ is a word, $\lambda_B$ is the mixture weight for the background model $\theta_B$, and $\pi_{d,i}$ is the document-specific mixture weight for the $i$-th cluster model $\theta_i$. We then estimate the cluster models by maximizing the probability of the pseudo-feedback documents being generated from the multinomial mixture model:

$$\log p(D|\Lambda) = \sum_{d \in D} \sum_{w \in V} c(w; d) \log p(w|d)$$

where $D = \{d_i | i = 1, 2, \cdots N\}$ is the set of the $N$ documents, $V$ is the vocabulary, $c(w; d)$ is $w$'s frequency in $d$ and $\Lambda = \{\theta_i | i = 1, 2, \cdots K\} \cup \{\pi_{d_i j} | i = 1, 2, \cdots N, j = 1, 2, \cdots K\}$ is the set of model parameters to estimate. The cluster models can be efficiently estimated using the Expectation-Maximization (EM) algorithm. For its details, we refer the reader to [26]. Table 1 shows the cluster models for TREC query "Transportation tunnel disasters" ($K = 3$). Note that only the middle cluster is relevant.

**Table 1: Cluster models for topic 363 "Transportation tunnel disasters"**

| Cluster 1 | | Cluster 2 | | Cluster 3 | |
|---|---|---|---|---|---|
| tunnel | 0.0768 | tunnel | 0.0935 | tunnel | 0.0454 |
| transport | 0.0364 | fire | 0.0295 | transport | 0.0406 |
| traffic | 0.0206 | truck | 0.0236 | toll | 0.0166 |
| railwai | 0.0186 | french | 0.0220 | amtrak | 0.0153 |
| harbor | 0.0146 | smoke | 0.0157 | train | 0.0129 |
| rail | 0.0140 | car | 0.0154 | airport | 0.0122 |
| bridg | 0.0139 | italian | 0.0152 | turnpik | 0.0105 |
| kilomet | 0.0136 | firefight | 0.0144 | lui | 0.0095 |
| truck | 0.0133 | blaze | 0.0127 | jersei | 0.0093 |
| construct | 0.0131 | blanc | 0.0121 | pass | 0.0087 |
| ... | | ... | | ... | |

From each of the $K$ estimated clusters, we choose the $L = M/K$ terms with highest probabilities to form a total of $M$ presentation terms. If a term happens to be in top $L$ in multiple clusters, we assign it to the cluster where it has highest probability and let the other clusters take one more term as compensation. We also filter out terms in the original query text because they tend to always be relevant when the query is short. The selected terms are then presented to the user for judgment. A sample (completed) feedback form is shown in Figure 1.

In this study we only deal with binary judgment: a presented term is by default unchecked, and a user may check it to indicate relevance. We also do not explicitly exploit negative feedback (i.e., penalizing irrelevant terms), because with binary feedback an unchecked term is not necessarily irrelevant (maybe the user is unsure about its relevance). We could ask the user for finer judgment (e.g., choosing from *highly relevant*, *somewhat relevant*, *do not know*, *somewhat irrelevant* and *highly irrelevant*), but binary feedback is more compact, taking less space to display and less user effort to make judgment.

# 5. ESTIMATING QUERY MODELS FROM TERM FEEDBACK

In this section, we present several algorithms for exploiting term feedback. The algorithms take as input the original query $q$, the clusters $\{\theta_i\}$ as generated by the theme discovery algorithm, the set of feedback terms $T$ and their relevance judgment $R$, and outputs an updated query language model $\theta'_q$ that makes best use of the feedback evidence to capture the user's information need.

First we describe our notations:

- $\theta_q$: The original query model, derived from query terms only:

  $$p(w|\theta_q) = \frac{c(w; q)}{|q|}$$

  where $c(w; q)$ is the count of $w$ in $q$, and $|q| = \sum_{w \in q} c(w; q)$ is the query length.

- $\theta_{q'}$: The updated query model which we need to estimate from term feedback.

- $\theta_i$ $(i = 1, 2, \ldots K)$: The unigram language model of cluster $C_i$, as estimated using the theme discovery algorithm.

- $T = \{t_{i,j}\}$ $(i = 1 \ldots K, j = 1 \ldots L)$: The set of terms presented to the user for judgment. $t_{i,j}$ is the $j$-th term chosen from cluster $C_i$.

- $R = \{\delta_w | w \in T\}$: $\delta_w$ is an indicator variable that is 1 if $w$ is judged relevant or 0 otherwise.

## 5.1 TFB (Direct Term Feedback)

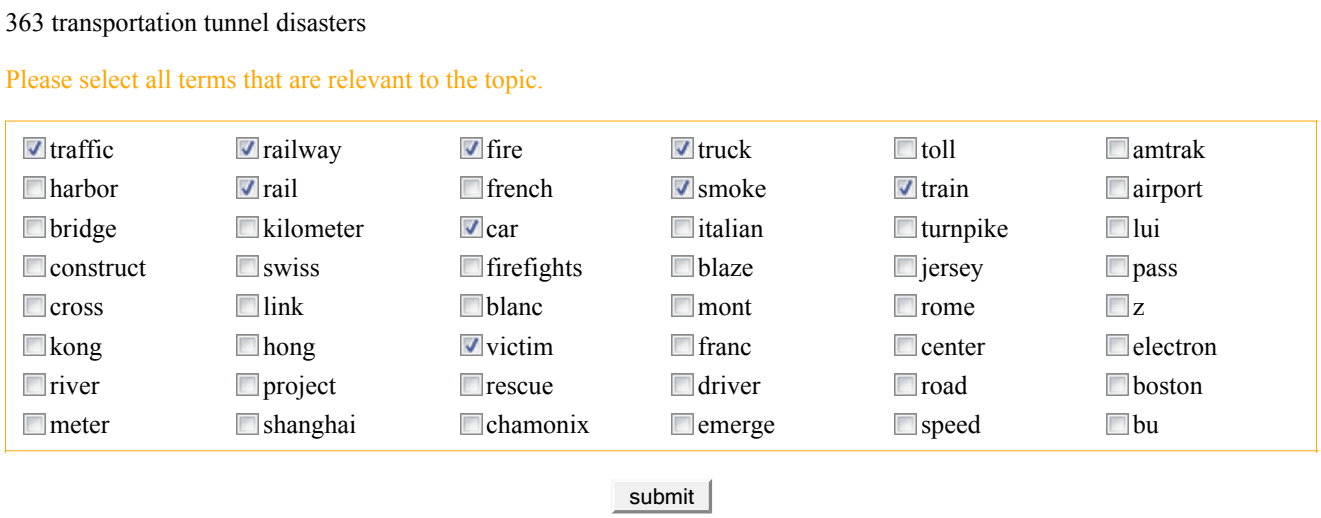
This is a straight-forward form of term feedback that does not involve any secondary structure. We give a weight of 1 to terms judged relevant by the user, a weight of $\mu$ to query terms, zero weight to other terms, and then apply normalization:

$$p(w|\theta_{q'}) = \frac{\delta_w + \mu\, c(w; q)}{\sum_{w' \in T} \delta_{w'} + \mu |q|}$$

where $\sum_{w' \in T} \delta_{w'}$ is the total number of terms that are judged relevant. We call this method TFB (direct Term FeedBack).

If we let $\mu = 1$, this approach is equivalent to appending the relevant terms after the original query, which is what standard query expansion (without term reweighting) does. If we set $\mu > 1$, we are putting more emphasis on the query terms than the checked ones. Note that the result model will be more biased toward $\theta_q$ if the original query is long or the user feedback is weak, which makes sense, as we can trust more on the original query in either case.

**Figure 1: Filled clarification form for Topic 363**




## 5.2 CFB (Cluster Feedback)

Here we exploit the cluster structure that played an important role when we selected the presentation terms. The clusters represent different aspects of the query topic, each of which may or may not be relevant. If we are able to identify the relevant clusters, we can combine them to generate a query model that is good at discovering documents belonging to these clusters (instead of the irrelevant ones). We could ask the user to directly judge the relevance of a cluster after viewing representative terms in that cluster, but this would sometimes be a difficult task for the user, who has to guess the semantics of a cluster via its set of terms, which may not be well connected to one another due to a lack of context. Therefore, we propose to learn cluster feedback *indirectly*, inferring the relevance of a cluster through the relevance of its feedback terms.

Because each cluster has an equal number of terms presented to the user, the simplest measure of a cluster's relevance is the number of terms that are judged relevant in it. Intuitively, the more terms are marked relevant in a cluster, the closer the cluster is to the query topic, and the more the cluster should participate in query modification. If we combine the cluster models using weights determined this way and then interpolate with the original query model, we get the following formula for query updating, which we call CFB (Cluster FeedBack):

$$p(w|\theta_{q'}) = \lambda p(w|\theta_q) + (1-\lambda)\sum_{i=1}^{K}\frac{\sum_{j=1}^{L}\delta_{t_{i,j}}}{\sum_{k=1}^{K}\sum_{j=1}^{L}\delta_{t_{k,j}}}p(w|\theta_i)$$

where $\sum_{j=1}^{L}\delta_{t_{i,j}}$ is the number of relevant terms in cluster $C_i$, and $\sum_{k=1}^{K}\sum_{j=1}^{L}\delta_{t_{k,j}}$ is the total number of relevant terms.

We note that when there is only one cluster ($K = 1$), the above formula degenerates to

$$p(w|\theta_{q'}) = \lambda p(w|\theta_q) + (1-\lambda)p(w|\theta_1)$$

which is merely pseudo-feedback of the form proposed in [25].

## 5.3 TCFB (Term-cluster Feedback)

TFB and CFB both have their drawbacks. TFB assigns non-zero probabilities to the presented terms that are marked relevant, but completely ignores (a lot more) others, which may be left unchecked due to the user's ignorance, or simply not included in the presentation list, but we should be able to infer their relevance from the checked ones. For example, in Figure 1, since as many as 5 terms in the middle cluster (the third and fourth columns) are checked, we should have high confidence in the relevance of other terms in that cluster. CFB remedies TFB's problem by treating the terms in a cluster collectively, so that unchecked/unpresented terms receive weights when presented terms in their clusters are judged as relevant, but it does not distinguish which terms in a cluster are presented or judged. Intuitively, the judged relevant terms should receive larger weights because they are explicitly indicated as relevant by the user. Therefore, we try to combine the two methods, hoping to get the best out of both.

We do this by interpolating the TFB model with the CFB model, and call it TCFB:

$$p(w|\theta_{q'}) = \alpha p(w|\theta_{q'_{TFB}}) + (1-\alpha)p(w|\theta_{q'_{CFB}})$$

# 6. EXPERIMENTS

In this section, we describe our experiment results. We first describe our experiment setup and present an overview of various methods' performance. Then we discuss the effects of varying the parameter setting in the algorithms, as well as the number of presentation terms. Next we analyze user term feedback behavior and its relation to retrieval performance. Finally we compare term feedback to relevance feedback and show that it has its particular advantage.

## 6.1 Experiment Setup and Basic Results

We took the opportunity of TREC 2005 HARD Track[2] for the evaluation of our algorithms. The tracks used the AQUAINT collection, a 3GB corpus of English newswire text. The topics included 50 ones previously known to be hard, i.e. with low retrieval performance. It is for these hard topics that user feedback is most helpful, as it can provide information to disambiguate the queries; with easy topics the user may be unwilling to spend efforts for feedback if the automatic retrieval results are good enough. Participants of the track were able to submit custom-designed *clarification forms* (CF) to solicit feedback from human assessors provided by

**Table 2: Retrieval performance for different methods and CF types. The last row is the percentage of MAP improvement over the baseline. The parameter settings $\mu = 4$, $\lambda = 0.1$, $\alpha = 0.3$ are near optimal.**

| | Baseline | TFB1C | TFB3C | TFB6C | CFB1C | CFB3C | CFB6C | TCFB1C | TCFB3C | TCFB6C |
|---|---|---|---|---|---|---|---|---|---|---|
| MAP | 0.219 | 0.288 | 0.288 | 0.278 | 0.254 | 0.305 | 0.301 | 0.274 | **0.309** | 0.304 |
| Pr@30 | 0.393 | 0.467 | 0.475 | 0.457 | 0.399 | 0.480 | 0.473 | 0.431 | **0.491** | 0.473 |
| RR | 4339 | 4753 | 4762 | 4740 | 4600 | 4907 | 4872 | 4767 | **4947** | 4906 |
| % | 0% | 31.5% | 31.5% | 26.9% | 16.0% | 39.3% | 37.4% | 25.1% | **41.1%** | 38.8% |

**Table 3: MAP variation with the number of presented terms.**

| # terms | TFB1C | TFB3C | TFB6C | CFB3C | CFB6C | TCFB3C | TCFB6C |
|---|---|---|---|---|---|---|---|
| 6 | 0.245 | 0.240 | 0.227 | 0.279 | 0.279 | 0.281 | 0.274 |
| 12 | 0.261 | 0.261 | 0.242 | 0.299 | 0.286 | 0.297 | 0.281 |
| 18 | 0.275 | 0.274 | 0.256 | 0.301 | 0.282 | 0.300 | 0.286 |
| 24 | 0.276 | 0.281 | 0.265 | 0.303 | 0.292 | 0.305 | 0.292 |
| 30 | 0.280 | 0.285 | 0.270 | 0.304 | 0.296 | 0.307 | 0.296 |
| 36 | 0.282 | 0.288 | 0.272 | 0.307 | 0.297 | 0.309 | 0.297 |
| 42 | 0.283 | 0.288 | 0.275 | 0.306 | 0.298 | 0.309 | 0.300 |
| 48 | 0.288 | 0.288 | 0.278 | 0.305 | 0.301 | 0.309 | 0.303 |

NIST. We designed three sets of clarification forms for term feedback, differing in the choice of $K$, the number of clusters, and $L$, the number of presented terms from each cluster. They are: $1 \times 48$, a big cluster with 48 terms, $3 \times 16$, 3 clusters with 16 terms each, and $6 \times 8$, 6 clusters with 8 terms each. The total number of presented terms (M) is fixed at 48, so by comparing the performance of different types of clarification forms we can know the effects of different degree of clustering. For each topic, an assessor would complete the forms ordered by $6 \times 8$, $1 \times 48$ and $3 \times 16$, spending up to three minutes on each form. The sample clarification form shown in Figure 1 is of type $3 \times 16$. It is a simple and compact interface in which the user can check relevant terms. The form is self-explanatory; there is no need for extra user training on how to use it.

Our initinal queries are constructed only using the topic title descriptions, which are on average 2.7 words in length. As our baseline we use the KL divergence retrieval method implemented in the Lemur Toolkit[1] with 5 pseudo-feedback documents. We stem the terms, choose Dirichlet smoothing with a prior of 2000, and truncate query language models to 50 terms (these settings are used throughout the experiments). For all other parameters we use Lemur's default settings. The baseline turns out to perform above average among the track participants. After an initial run using this baseline retrieval method, we take the top 60 documents for each topic and apply the theme discovery algorithm to output the clusters (1, 3, or 6 of them), based on which we generate clarification forms. After user feedback is received, we run the term feedback algorithms (TFB, CFB or TCFB) to estimate updated query models, which are then used for a second iteration of retrieval.

We evaluate the different retrieval methods' performance on their rankings of the top 1000 documents. The evaluation metrics we adopt include mean average (non-interpolated) precision (MAP), precision at top 30 (Pr@30) and total relevant retrieved (RR). Table 2 shows the performance of various methods and configurations of $K \times L$. The suffixes (1C, 3C, 6C) after TFB,CFB,TCFB stand for the number of clusters ($K$). For example, TCFB3C means the TCFB method on the $3 \times 16$ clarification forms.

From Table 2 we can make the following observations:

1. All methods perform considerably better than the pseudo-feedback baseline, with TCFB3C achieving a highest 41.1% improvement in MAP, indicating significant contribution of term feedback for clarification of the user's information need. In other words, term feedback is truly helpful for improving retrieval accuracy.

2. For TFB, the performance is almost equal on the $1 \times 48$ and $3 \times 16$ clarification forms in terms of MAP (although the latter is slightly better in Pr@30 and RR), and a little worse on the $6 \times 8$ ones.

3. Both CFB3C and CFB6C perform better than their TFB counterparts in all three metrics, suggesting that feedback on a secondary cluster structure is indeed beneficial. CFB1C is actually worse because it cannot adjust the weight of its (single) cluster from term feedback and it is merely pseudo-feedback.

4. Although TCFB is just a simple mixture of TFB and CFB by interpolation, it is able to outperform both. This supports our speculation that TCFB overcomes the drawbacks of TFB (paying attention only to checked terms) and CFB (not distinguishing checked and unchecked terms in a cluster). Except for TCFB6C v.s. CFB6C, the performance advantage of TCFB over TFB/CFB is significant at $p < 0.05$ using the Wilcoxon signed rank test. This is not true in the case of TFB v.s. CFB, each of which is better than the other in nearly half of the topics.

## 6.2 Reduction of Presentation Terms

In some situations we may have to reduce the number of presentation terms due to limits in display space or user feedback efforts. It is interesting to know whether our algorithms' performance deteriorates when the user is presented with fewer terms. Because the presentation terms within each cluster are generated in decreasing order of their frequencies, the presentation list forms a subset of the original one if its size is reduced[2]. Therefore, we can easily simulate what happens when the number of presentation terms decreases

[1] http://www.lemurproject.com

[2] There are complexities arising from terms appearing in top $L$ of multiple clusters, but these are exceptions

from $M$ to $M'$: we will keep all judgments of the top $L' = M'/K$ terms in each cluster and discard those of others. Table 3 shows the performance of various algorithms as the number of presentation terms ranges from 6 to 48.

We find that the performance of TFB is more susceptible to presentation term reduction than that of CFB or TCFB. For example, at 12 terms the MAP of TFB3C is 90.6% of that at 48 terms, while the numbers for CFB3C and TCFB3C are 98.0% and 96.1% respectively. We conjecture the reason to be that while TFB's performance heavily depends on how many good terms are chosen for query expansion, CFB only needs a rough estimate of cluster weights to work. Also, the $3 \times 16$ clarification forms seem to be more robust than the $6 \times 8$ ones: at 12 terms the MAP of TFB6C is 87.1% of that at 48 terms, lower than 90.6% for TFB3C. Similarly, for CFB it is 95.0% against 98.0%. This is natual, as for a large cluster number of 6, it is easier to get into the situation where each cluster gets too few presentation terms to make topic diversification useful.

Overall, we are surprised to see that the algorithms are still able to perform reasonably well when the number of presentation terms is small. For example, at only 12 terms CFB3C (the clarification form is of size $3 \times 4$) can still improve 36.5% over the baseline, dropping slightly from 39.3% at 48 terms.

## 6.3 User Feedback Analysis

In this part we study several aspects of user's term feedback behavior, and whether they are connected to retrieval performance.

**Figure 2: Clarification form completion time distributions**

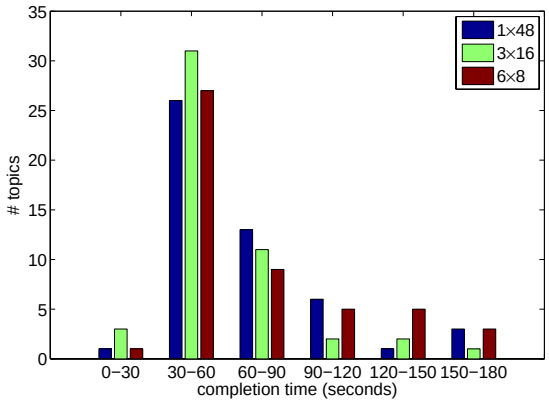



Figure 2 shows the distribution of time needed to complete a clarification form[3]. We see that the user is usually able to finish term feedback within a reasonably short amount of time: for more than half of the topics the clarification form is completed in just 1 minute, and only a small fraction of topics (less than 10% for $1 \times 48$ and $3 \times 16$) take more than 2 minutes. This suggests that term feedback is suitable for interactive ad-hoc retrieval, where a user usually does not want to spend too much time on providing feedback.

We find that a user often makes mistakes when judging term relevance. Sometimes a relevant term may be left out because its connection to the query topic is not obvious to the user. Other times a dubious term may be included but turns out to be irrelevant. Take the topic in Figure 1 for example. There was a fire disaster in Mont Blanc Tunnel between France and Italy in 1999, but the user failed to select such keywords as "mont", "blanc", "french" and "italian" due to his/her ignorance of the event. Indeed, without proper context it would be hard to make perfect judgment.

[3]The maximal time is 180 seconds, as the NIST assessor would be forced to submit the form at that moment.

**Table 4: Term selection statistics (topic average)**

| CF Type | $1 \times 48$ | $3 \times 16$ | $6 \times 8$ |
|---|---|---|---|
| # checked terms | 14.8 | 13.3 | 11.2 |
| # rel. terms | 15.0 | 12.6 | 11.2 |
| # rel. checked terms | 7.9 | 6.9 | 5.9 |
| precision | 0.534 | 0.519 | 0.527 |
| recall | 0.526 | 0.548 | 0.527 |

What is then, the extent to which the user is good at term feedback? Does it have serious impact on retrieval performance? To answer these questions, we need a measure of individual terms' true relevance. We adopt the Simplified KL Divergence metric used in [24] to decide query expansion terms as our term relevance measure:

$$\sigma_{KLD}(w) = p(w|R) \log \frac{p(w|R)}{p(w|\neg R)}$$

where $p(w|R)$ is the probability that a relevant document contains term $w$, and $p(w|\neg R)$ is the probability that an irrelevant document contains $w$, both of which can be easily computed via maximum likelihood estimate given document-level relevance judgment. If $\sigma_{KLD}(w) > 0$, $w$ is more likely to appear in relevant documents than irrelevant ones.

We consider a term relevant if its Simplified KL Divergence value is greater than a certain threshold $\sigma_0$. We can then define precision and recall of user term judgment accordingly: precision is the fraction of terms checked by the user that are relevant; recall is the fraction of presented relevant terms that are checked by the user. Table 4 shows the number of checked terms, relevant terms and relevant checked terms when $\sigma_0$ is set to 1.0, as well as the precision/recall of user term judgment.

Note that when the clarification forms contain more clusters, fewer terms are checked: 14.8 for $1 \times 48$, 13.3 for $3 \times 16$ and 11.2 for $6 \times 8$. Similar pattern holds for relevant terms and relevant checked terms. There seems to be a trade-off between increasing topic diversity by clustering and losing extra relevant terms: when there are more clusters, each of them gets fewer terms to present, which can hurt a major relevant cluster that contains many relevant terms. Therefore, it is not always helpful to have more clusters, e.g., TFB6C is actually worse than TFB1C.

The major finding we can make from Table 4 is that the user is not particularly good at identifying relevant terms, which echoes the discovery in [18]. In the case of $3 \times 16$ clarification forms, the average number of terms checked as relevant by the user is 13.3 per topic, and the average number of relevant terms whose $\sigma_{KLD}$ value exceed 1.0 is 12.6. The user is able to recognize only 6.9 of these terms on average. Indeed, the precision and recall of user feedback terms (as defined previously) are far from perfect. On the other hand, If the user had correctly checked all such relevant terms, the performance of our algorithms would have increased a lot, as shown in Table 5.

We see that TFB gets big improvement when there is an oracle who checks all relevant terms, while CFB meets a bottleneck around MAP of 0.325, since all it does is adjust cluster weights, and when the learned weights are close to being accurate, it cannot benefit more from term feedback. Also note that TCFB fails to outperform TFB, probably because TFB is sufficiently accurate.

**Table 5: Change of MAP when using all (and only) relevant terms ($\sigma_{KLD} > 1.0$) for feedback.**

| | original term feedback | relevant term feedback |
|---|---|---|
| TF1 | 0.288 | 0.354 |
| TF3 | 0.288 | 0.354 |
| TF6 | 0.278 | 0.346 |
| CF3 | 0.305 | 0.325 |
| CF6 | 0.301 | 0.326 |
| TCF3 | 0.309 | 0.345 |
| TCF6 | 0.304 | 0.341 |

## 6.4 Comparison with Relevance Feedback

Now we compare term feedback with document-level relevance feedback, in which the user is presented with the top $N$ documents from an initial retrieval and asked to judge their relevance. The feedback process is simulated using document relevance judgment from NIST. We use the mixture model based feedback method proposed in [25], with mixture noise set to 0.95 and feedback coefficient set to 0.9.

Comparative evaluation of relevance feedback against other methods is complicated by the fact that some documents have already been viewed during feedback, so it makes no sense to include them in the retrieval results of the second run. However, this does not hold for term feedback. Thus, to make it fair w.r.t. user's information gain, if the feedback documents are relevant, they should be kept in the top of the ranking; if they are irrelevant, they should be left out. Therefore, we use relevance feedback to produce a ranking of top 1000 retrieved documents but with every feedback document excluded, and then prepend the relevant feedback documents at the front. Table 6 shows the performance of relevance feedback for different values of $N$ and compares it with TCFB3C.

**Table 6: Performance of relevance feedback for different number of feedback documents ($N$).**

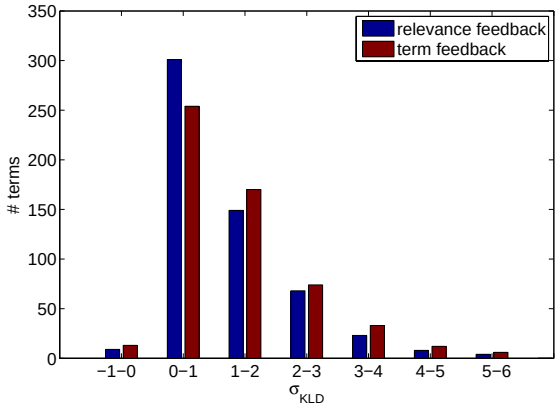
| $N$ | MAP | Pr@30 | RR |
|---|---|---|---|
| 5 | 0.302 | 0.586 | 4779 |
| 10 | 0.345 | 0.670 | 4916 |
| 20 | 0.389 | 0.772 | 5004 |
| TCFB3C | 0.309 | 0.491 | 4947 |

We see that the performance of TCFB3C is comparable to that of relevance feedback using 5 documents. Although it is poorer than when there are 10 feedback documents in terms of MAP and Pr@30, it does retrieve more documents (4947) when going down the ranked list.

We try to compare the quality of automatically inserted terms in relevance feedback with that of manually selected terms in term feedback. This is done by truncating the relevance feedback modified query model to a size equal to the number of checked terms for the same topic. We can then compare the terms in the truncated model with the checked terms. Figure 3 shows the distribution of the terms' $\sigma_{KLD}$ scores.

We find that term feedback tends to produce expansion terms of higher quality(those with $\sigma_{KLD} > 1$) compared to relevance feedback (with 10 feedback documents). This does not contradict the fact that the latter yields higher retrieval performance. Actually, when we use the truncated query model instead of the intact one refined from relevance feedback, the MAP is only 0.304. The truth

**Figure 3: Comparison of expansion term quality between relevance feedback (with 10 feedback documents) and term feedback (with $3 \times 16$ CFs)**




is, although there are many unwanted terms in the expanded query model from feedback documents, there are also *more* relevant terms than what the user can possibly select from the list of presentation terms generated with pseudo-feedback documents, and the positive effects often outweighs the negative ones.

We are interested to know under what circumstances term feedback has advantage over relevance feedback. One such situation is when *none* of the top $N$ feedback documents is relevant, rendering relevance feedback useless. This is *not* infrequent, as one might have thought: out of the 50 topics, there are 13 such cases when $N = 5$, 10 when $N = 10$, and still 3 when $N = 20$. When this happens, one can only back off to the original retrieval method; the power of relevance feedback is lost.

Surprisingly, in 11 out of 13 such cases where relevance feedback seems impossible, the user is able to check at least 2 relevant terms from the $3 \times 16$ clarification form (we consider term $t$ to be relevant if $\sigma_{KLD}(t) > 1.0$). Furthermore, in 10 out of them TCFB3C outperforms the pseudo-feedback baseline, increasing MAP from 0.076 to 0.146 on average (these are particularly hard topics). We think that there are two possible explanations for this phenomenon of term feedback being active even when relevance feedback does not work: First, even if none of the top $N$ (suppose it is a small number) documents are relevant, we may still find relevant documents in top 60, which is more inclusive but usually unreachable when people are doing relevance feedback in interactive ad-hoc search, from which we can draw feedback terms. This is true for topic 367 "piracy", where the top 10 feedback documents are all about software piracy, yet there are documents between 10-60 that are about piracy on the seas (which is about the real information need), contributing terms such as "pirate", "ship" for selection in the clarification form. Second, for some topics, a document needs to meet some special condition in order to be relevant. The top $N$ documents may be related to the topic, but nonetheless irrelevant. In this case, we may still extract useful terms from these documents, even if they do not qualify as relevant ones. For example, in topic 639 "consumer online shopping", a document needs to mention what contributes to shopping growth to really match the specified information need, hence none of the top 10 feedback documents are regarded as relevant. But nevertheless, the feedback terms such as "retail", "commerce" are good for query expansion.

## 7. CONCLUSIONS

In this paper we studied the use of term feedback for interactive information retrieval in the language modeling approach. We proposed a cluster-based method for selecting presentation terms as well as algorithms to estimate refined query models from user term feedback. We saw significant improvement in retrieval accuracy brought by term feedback, in spite of the fact that a user often makes mistakes in relevance judgment that hurts its performance. We found the best-performing algorithm to be TCFB, which benefits from the combination of directly observed term evidence with TFB and indirectly learned cluster relevance with CFB. When we reduced the number of presentation terms, term feedback is still able to keep much of its performance gain over the baseline. Finally, we compared term feedback to document-level relevance feedback, and found that TCFB3C's performance is on a par with the latter with 5 feedback documents. We regarded term feedback as a viable alternative to traditional relevance feedback, especially when there are no relevant documents in the top.

We propose to extend our work in several ways. First, we want to study whether the use of various contexts can help the user to better identify term relevance, while not sacrificing the simplicity and compactness of term feedback. Second, currently all terms are presented to the user in a single batch. We could instead consider iterative term feedback, by presenting a small number of terms first, and show more terms after receiving user feedback or stop when the refined query is good enough. The presented terms should be selected *dynamically* to maximize learning benefits at any moment. Third, we have plans to incorporate term feedback into our UCAIR toolbar[20], an Internet Explorer plugin, to make it work for web search. We are also interested in studying how to combine term feedback with relevance feedback or implicit feedback. We could, for example, allow the user to dynamically modify terms in a language model learned from feedback documents.

## 8. ACKNOWLEDGMENT

This work is supported in part by the National Science Foundation grants IIS-0347933 and IIS-0428472.

## 9. REFERENCES

[1] J. Allan. Relevance feedback with too much data. In *Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval*, pages 337–343, 1995.

[2] J. Allan. HARD track overview in TREC 2005 – High Accuracy Retrieval from Documents. In *The Fourteenth Text REtrieval Conference*, 2005.

[3] P. Anick. Using terminological feedback for web search refinement: a log-based study. In *Proceedings of the 26th annual international ACM SIGIR conference on research and development in informaion retrieval*, pages 88–95, 2003.

[4] P. G. Anick and S. Tipirneni. The paraphrase search assistant: terminological feedback for iterative information seeking. In *Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval*, pages 153–159, 1999.

[5] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic query expansion using SMART. In *Proceedings of the Third Text REtrieval Conference*, 1994.

[6] D. Harman. Towards interactive query expansion. In *Proceedings of the 11th annual international ACM SIGIR conference on research and development in information retrieval*, pages 321–331, 1988.

[7] N. A. Jaleel, A. Corrada-Emmanuel, Q. Li, X. Liu, C. Wade, and J. Allan. UMass at TREC 2003: HARD and QA. In *TREC*, pages 715–725, 2003.

[8] H. Joho, C. Coverson, M. Sanderson, and M. Beaulieu. Hierarchical presentation of expansion terms. In *Proceedings of the 2002 ACM symposium on applied computing*, pages 645–649, 2002.

[9] K. S. Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and status. Technical Report 446, Computer Laboratory, University of Cambridge, 1998.

[10] D. Kelly, V. D. Dollu, and X. Fu. The loquacious user: a document-independent source of terms for query expansion. In *Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval*, pages 457–464, 2005.

[11] D. Kelly and X. Fu. Elicitation of term relevance feedback: an investigation of term source and context. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval*, 2006.

[12] J. Koenemann and N. Belkin. A case for interaction: A study of interactive information retrieval behavior and effectiveness. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 205–212, 1996.

[13] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Research and Development in Information Retrieval*, pages 120–127, 2001.

[14] Y. Nemeth, B. Shapira, and M. Taeib-Maimon. Evaluation of the real and perceived value of automatic and interactive query expansion. In *Proceedings of the 27th annual international ACM SIGIR conference on research and development in information retrieval*, pages 526–527, 2004.

[15] J. Ponte. *A Language Modeling Approach to Information Retrieval*. PhD thesis, University of Massachusetts at Amherst, 1998.

[16] S. E. Robertson, S. Walker, S. Jones, M. Beaulieu, and M. Gatford. Okapi at TREC-3. In *Proceedings of the Third Text REtrieval Conference*, 1994.

[17] J. Rocchio. Relevance feedback in information retrieval. In *The SMART retrieval system*, pages 313–323. 1971.

[18] I. Ruthven. Re-examining the potential effectiveness of interactive query expansion. In *Proceedings of the 26th annual international ACM SIGIR conference on research and development in informaion retrieval*, pages 213–220, 2003.

[19] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41:288–297, 1990.

[20] X. Shen, B. Tan, and C. Zhai. Implicit user modeling for personalized search. In *Proceedings of the 14th ACM international conference on information and knowledge management*, pages 824–831, 2005.

[21] X. Shen and C. Zhai. Active feedback in ad hoc information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval*, pages 59–66, 2005.

[22] A. Spink. Term relevance feedback and query expansion: relation to design. In *Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval*, pages 81–90, 1994.

[23] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval*, pages 4–11, 1996.

[24] H. Zaragoza, N. Craswell, M. Taylor, S. Saria, and S. Robertson. Microsoft cambridge at TREC-13: Web and HARD tracks. In *Proceedings of the 13th Text REtrieval Conference*, 2004.

[25] C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the tenth international conference on information and knowledge management*, pages 403–410, 2001.

[26] C. Zhai, A. Velivelli, and B. Yu. A cross-collection mixture model for comparative text mining. In *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining*, pages 743–748, 2004.