TREC-2006 at Maryland: Blog, Enterprise, Legal and QA Tracks

Douglas Oard,[§*] Tamer Elsayed,[†] Jianqiang Wang,[‡] Yejun Wu,[§]
Pengyi Zhang,[§] Eileen Abels,[¶] Jimmy Lin,[§*] and Dagbert Soergel[§]
University of Maryland, College Park, MD 20742
{oard,telsayed,wuyj,pengyi,jimmylin,dsoergel}@umd.edu

Abstract

In TREC 2006, teams from the University of Maryland participated in the Blog track, the Expert Search task of the Enterprise track, the Complex Interactive Question Answering task of the Question Answering track, and the Legal track. This paper reports our results.

1 Blog Track

Blogs are being hailed as fundamentally different from other Internet communication protocols (e.g., email, WWW), and as possessing a socially-transformative, democratizing potential [9]. Journalists see blogs as alternative sources of news and public opinion [12]. "Blogs tend to be impressionistic, telegraphic, raw, honest, individualistic, highly opinionated and passionate, often striking an emotional chord" [13]. Private individuals create blogs as a vehicle for self-expression and self-empowerment [3, 9]. Therefore the blogosphere is a huge information space of unstructured informal text in which opinions and attitudes are embedded.

People may have different information needs concerning opinions and attitudes. Political candidates may wish to know both the aggregate attitudes (or sentiments) toward them and which groups of people like/dislike them. Policy makers and journalists may want to know the whole spectrum of attitudes of stakeholders (including foreign countries) on an issue. Advertisers may want to know the change of aggregate attitudes after the advertisement is delivered to the targeted population. Individuals may want to know a certain celebrity's attitudes about an issue, or to find people who share their attitudes to have a discussion, or to find people who disagree with their attitudes to persuade them. Therefore opinion retrieval can be useful to many people.

Previously there have been opinion retrieval studies on small collections of news articles, blogs, and Web forums. However, the TREC-2006 Blog pilot track offered the first opportunity to create a large public test collection for evaluation of blog opinion search using a large collection of blogs. The pilot track has two tasks, a main task (opinion retrieval) which focuses on the opinionated nature of many blogs, and an open task which aims to determine a suitable second task for 2007 on other aspects of blogs. The University of Maryland participated in the opinion retrieval task, which involves locating blog posts that express an opinion about a given target. "The target can be a 'traditional' named entity – a name of a person, location, or organization – but also a concept (such as a type of technology), a product name, or an event."[1] In this first year of the track, NIST judges created 50 topics and performed relevance judgment. No training topics were available. Results from this year of the opinion retrieval track should therefore be considered preliminary—the main goal is to explore the design space for tasks and metrics and to create an initial test collection for use in formative evaluation.

1.1 Methods

In our TREC-2006 opinion retrieval experiments, we tested the following ideas: (1)segmenting permalink documents to natural paragraphs and fixed sized passages to examine their difference in retrieval effectiveness;

*Institute for Advanced Computer Studies, University of Maryland
†Computer Science Department, University of Maryland
‡Present Address: Department of Library and Information Science, State University of New York at Buffalo
§College of Information Studies, University of Maryland
¶Present Address: College of Information Science and Technology, Drexel University; E-mail: eileen.abels@ischool.drexel.edu
[1] *http://www.science.uva.nl/research/iiwiki/wiki/index.php/Guidelines_for_Participants_2006*

(2)demoting the non-opinionated paragraphs along the retrieved ranked list; (3)query formulation using the title fields only versus using both the title and the description fields.

The blog collection was crawled over a period of 11 weeks (December 2005 - February 2006). The total size of the collection amounts to 23 GB (compressed) with three main components: feeds (8 GB), permalinks (11 GB), and homepages (4 GB). The collection contains spam as well as possibly non-blogs, e.g. RSS feeds from news broadcasters. The distributed collection takes the form of a collection of organized, and uniquely identified XML feeds and their corresponding HTML blog posting pages.[2]

Since the retrieval units are the documents from the permalinks, we use the permalinks only. Each permalink file is basically a blog posting plus its comments (if any) embedded in a Web page with HTML markup. We used a perl program to remove javascript codes and then used the Lynx utility to strip off the HTML tags. Each html-stripped document contains a blog posting and possible comments surrounded by noisy information such as Web page navigation, advertising, and copyright. We sent out an email to the participants to call for a joint effort to clean the collection but received little passion and few responses. Since each blog hosting site has its own pattern of web page design, we by ourselves can clean only a very limited number of such Web pages. By taking a look at the permalinks.txt file under each of the 71 folders (from 20051206 to 20060221) which records the permalinks of blog postings and comments, we roughly estimated the frequency of each blog hosting site and selected the top 5 sites (i.e., livejournal.com, spaces.msn.com, xanga.com, blogs.msdn.com, 6-allthe.info) for further cleaning, which accounts for approximately 10% of the total number of permalinks. We then manually examined 100 documents from each of the 5 sites trying to find patterns for removing noisy information on the permalink Web pages. Here are some example patterns for cleaning livejournal.com:
– remove anything beneath "Log in now."
– remove anything between "Page Summary" and a date pattern such as "November 24th, 2005."
– remove anything between the following hint strings and a string starting with a *: "Previous Posts[:]", "Recent blog posts[:]", "Recent Posts[:]", "Categories[:]", "Recent News[:]", "Advertising[:]", "Blogs I read[:]"; "Documents I liked[:]." Here [:] means either a space or a colon.
– remove consecutive short lines (less than 41 characters) starting with *, +, #, o, or @ followed by a space.
The latter two patterns were also applied to the permalink Web pages of other than the top 5 hosting sites. These are rough cleaning steps in an attempt to remove the titles of other posts, so there is still noisy information remaining after these steps. The size of the original permalinks and processed permalink files is listed in Table 1.

original permalinks	90.1GB (3215171 documents)
after stripping off HTML tags	22.1 GB
after further cleaning	17.9 GB

Table 1: Size of the Permalink Collection.

We think a whole permalink document may include attitudes toward multiple targets, so the ideal document granularity for attitude analysis is a paragraph or a passage. Paragraph detection is a text classification task which requires training data with boundaries marked. Moreover, blog postings and comments are all personal writings which have not gone through any professional editing, so automatic paragraph detection can be a challenging research problem here. To simplify the complicated problem, we manually examined a random sample of 200 permalink documents to find heuristic patterns for segmenting a document into natural non-overlapped paragraphs. Our algorithm consists of two steps. The first step identifies candidate paragraphs by merging lines. A current line is merged into a previous line if:
(1) the current line is an empty line or a line ending with any of the following punctuation marks: !, ., ?, ", and) if the previous line is not as such, or
(2) the previous line starts with any of the following symbols: -, >, *, +, o, ., or numbers. These symbols usually represent the title of a section.

The second step merges a current candidate paragraph into a previous candidate paragraph if:
(1) the merged paragraph has < 50 tokens, or
(2) the current paragraph has $>= 50$ tokens but the previous one has < 30 tokens, or
(3) the current paragraph has < 50 tokens and the merged paragraph has between 50 and 60 tokens, or
(4) the previous paragraph has $>= 50$ tokens, and the current paragraph has < 10 tokens, and the current

[2] $http://www.science.uva.nl/research/iiwiki/wiki/index.php/Guidelines\_for\_Participants\_2006$

paragraph does not start with the pattern of somebody "wrote:", "write:", "writes:", "said:", "say:", "says:", or (5) the previous paragraph has >= 50 tokens, the current paragraph has >= 10 tokens, the current paragraph does not start with the pattern of somebody "wrote:", "write:", "writes:", "said:", "say:", "says:", and each line of the current paragraph has same number of starting empty spaces as the previous paragraph.

The algorithm is still flawed since some of the generated paragraphs are very long, especially for those postings and comments which are ill-formatted, but we did not have enough time to improve it.

An alternative to segmenting a document into paragraph is to segment a document into overlapped fixed sized passages. Text tiling segments text into multi-paragraph subtopic passages [8], but is not appropriate here since it sets passage boundary when the topic changes, that is, it may fail to detect an appropriate passage for an attitude target when the topic and the target do not match. We generated overlapped fixed sized passages with window size of 50 words and overlap of 10 words between two neighbor windows.

We indexed the paragraphs and passages generated from the cleaned permalink collection using Indri[3]. The Porter stemmer was applied and stop words removed during indexing. Topics were processed into Indri queries with stop words removed. Indri automatically stems the query words using the stemmer specified during the indexing process. Two types of queries were formulated - using the title fields only, and using both the title and description fields. The relevant paragraphs or passages were retrieved for opinion analysis.

We took two steps to analyze the attitudes toward the attitude targets. First we computed the semantic orientations of the words in a relevant paragraph or passage, then aggregated the semantic orientations of the words of the relevant paragraph or passage to get a final sentiment score of the paragraph or passage. At first we collected the prior-polarity subjectivity lexicon from Wilson [20] which annotates the semantic orientation (i.e., positive, negative, neutral) of 8221 words.

Turney and Littman [18] presented an unsupervised learning method for inferring semantic orientation of words (including adjectives, adverbs, nouns, and verbs) from semantic association. Seven positive words (good, nice, excellent, positive, fortunate, correct, and superior) and 7 negative words (bad, nasty, poor, negative, unfortunate, correct, and superior) were intuitively selected as paradigms of positive and negative semantic orientation due to their lack of sensitivity to context. The semantic orientation of a given word was calculated from the strength of its association with the 7 positive words, minus the strength of its association with the 7 negative words. The magnitude of the difference can be considered as the strength of the semantic orientation. The strength of the semantic association between words were computed using two methods—pointwise mutual information (PMI) and latent semantic analysis. PMI was computed using word co-occurrence collected from the web search engine Altavista (i.e., the hits of a query "*word1* NEAR *word2*") as follows:

$$PMI(word_1, word_2) = log_2(\frac{\frac{1}{N}hits(word_1 NEAR word_2)}{\frac{1}{N}hits(word_1)\frac{1}{N}hits(word_2)}) \quad (1)$$

where N is the total number of documents indexed by the search engine. The PMI method, given an unlabeled Web training corpus of approximately one hundred billion words, attained an accuracy of 82.8% which is comparable with Hatzivassiloglou and McKeown's complex method [7].

Due to the time constraint, we were not allowed to compute the semantic orientations of all the words in the retrieved paragraphs or passages of all the 5 runs. However, Wiebe et al. found that subjectivity clues include low-frequency words, collocations, and adjectives and verbs identified using distributional similarity [19], so we selected the low frequency words from the top 1000 retrieved paragraphs using the title fields only. Since the method of counting positive and negative terms requires us to transform the terms into their base forms (lemmas) in order to be able to check if a term is in our sentiment lexicon [10], before selecting low frequency words, we lemmatized all the words of the retrieved paragraphs using Morpha[4], then removed stop words, the words in Wilson's subjectivity lexicon, and those not in a dictionary detected by the unix "spell" utility, then we selected the lemma with document frequency no more than 40 times ($DF <= 40$). We noticed that when $DF > 40$ more non-subjective lemmas were seen. Finally we selected 16773 words.

Since Altavista is not available any more and it is very time consuming to consult any Web search engine for word co-occurrence frequency, we computed the PMIs of the 16773 words by consulting the blog permalinks collections directly. We used Indri's RunQuery to collect hits of words and word co-occurrence. Whenever RunQuery reported a core dump for any word (due to too many retrieved documents), that word was removed. Nine words were removed due to this reason.

[3]http://www.lemurproject.org/indri/

[4]http://www.cogs.susx.ac.uk/research/nlp/carroll/morph.html

We aggregated a sentiment score of a paragraph or passage from the semantic orientations of the lemmas by checking two sources. If a lemma was in Wilson's subjectivity lemma lexicon, it got 1 point. Otherwise, if it was in the low frequency word list with a score between -3 and -0.05, or between 0.05 and 5, it got 0.2 point. Lemmas with a score of (-0.05, 0.05) were considered neutral, and those with a score (-infinite, -3) or (5, infinite) were considered abnormal, and therefore were removed. The aggregated sentiment score of a paragraph or passage is computed as a score accumulated from all the lemmas in the paragraph/passage normalized by the paragraph/passage length (i.e., the number of lemmas in the paragraph/passage).

We performed a proportional demotion of paragraphs/passages along the retrieved ranked list if a paragraph/passage has a normalized sentiment score less than 0.15. We tried two demotions - demotion by 2 times and 3 times. By demoting n times we mean demoting a paragraph/passage from its original position of x to the position of nx along the ranked list. Before submitting our runs, we merged the paragraphs/passages from a same permalink document back into one document bottom up along the ranked list. If fewer than 1000 documents were generated, we re-ran our queries to retrieve more paragraphs/passages.

1.2 Relevance Assessment

TREC organizes assessments for the opinion retrieval task. For the assessment, the content of a blog post is defined as the content of the post itself and the contents of all comments to the post; if the relevant content is in a comment, then the permalink is declared to be relevant. The assessors assigned 6 levels of relevance scores[5]: not judged (-1), irrelevant (0), relevant but non-opinionated (1), relevant with negative opinion (2), relevant with mixed (or ambiguous) opinion (3), relevant with positive opinion (4). Note that for opinion relevance, the topic of the post does not necessarily have to be the target, but an opinion about the target must be present in the post or one of the comments to the post. We report our results at two levels: topic relevant level ($>= 1$ vs < 1) and opinion relevant level ($>= 2$ vs < 2).

1.3 Results and Discussion

We submitted 5 automatic runs consisting of the top 1,000 documents for each topic for the opinion retrieval task.

ParTitDef. Baseline 1. This is our required automatic run. Permalink documents are segmented into paragraphs. Queries are automatically formulated using the title fields only. No opinion detection is performed.

ParTitDesDef. Baseline 2. Permalink documents are segmented into paragraphs. Queries are automatically formulated using both the title and description fields. No opinion detection is performed.

PasTitDesDef. Baseline 3. Permalink documents are segmented into passages. Queries are automatically formulated using both the title and description fields. No opinion detection is performed.

ParTiDesDmt2. Permalink documents are segmented into paragraphs. Queries are automatically formulated using both the title and description fields. Non-opinionated paragraphs are demoted 2 times proportionally along the retrieved ranked list.

ParTiDesDmt3. Permalink documents are segmented into paragraphs. Queries are automatically formulated using both the title and description fields. Non-opinionated paragraphs are demoted 3 times proportionally along the retrieved ranked list.

Our analysis to date has focused on retrieval effectiveness for topic relevance and opinion relevance. Table 2 shows retrieval effectiveness measures at topic relevance. Comparing ParTitDesDef with other four runs, we see a minor reduction of Mean uninterpolated Average Precision (MAP) for demotion of opinionated paragraphs (ParTiDesDmt2 and ParTiDesDmt3) and substantial reduction of MAP for ParTiDef and PasTiDesDef.

A Wilcoxon signed-rank test for paired samples indicated that the reduction in MAP between ParTitDesDef and ParTiDesDmt2 is not significant at $p < 0.05$, between ParTitDesDef and ParTiDesDmt3 is marginally significant at $p = 0.05$, between ParTitDesDef and ParTiDef or PasTiDesDef is significant at $p < 0.05$.

Table 3 shows retrieval effectiveness measure at opinion relevance. Comparing ParTitDesDef with other four runs, we see similar reduction patterns as Table 2 except a minor improvement of ParTiDesDmt2 over

[5] $http://www.science.uva.nl/research/iiwiki/wiki/index.php/Guidelines\_for\_Participants\_2006$

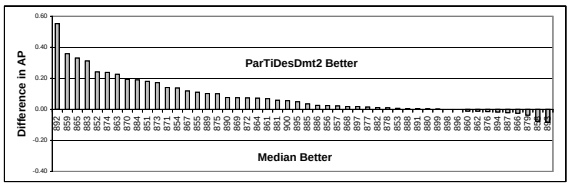

Figure 1: [Blog] Difference between ParTiDesDmt2 and Median, Average Precision at Opinion Relevance.

ParTitDesDef. A Wilcoxon signed-rank test for paired samples indicated that the difference in MAP between ParTitDesDef and ParTiDesDmt2 or ParTiDesDmt3 is not significant at $p < 0.05$, between ParTitDesDef and ParTiDef or PasTiDesDef is significant at $p < 0.05$.

From these comparisons we conclude that demotion of opinionated paragraphs does not help better retrieve either topically relevant or opinion relevant documents; query formulation using both the title and description fields is significantly better than using the title fields only; segmenting permalink documents into paragraphs is significantly better than segmenting into fixed sized passages.

Runs	MAP	Bpref	P10	R-Prec
ParTitDesDef	**0.2849**	0.3998	**0.6200**	0.3490
ParTiDesDmt2	0.2845	**0.4040**	**0.6200**	0.3501
ParTiDesDmt3	0.2812	0.4034	**0.6200**	**0.3542**
ParTiDef	0.2362	0.3580	0.5280	0.3162
PasTiDesDef	0.2733	0.3866	0.5800	0.3516

Table 2: Comparison at Topic Relevance.

Runs	MAP	Bpref	P10	R-Prec
ParTitDesDef	0.1882	0.2521	**0.3780**	**0.2441**
ParTiDesDmt2	**0.1887**	**0.2573**	**0.3780**	0.2421
ParTiDesDmt3	0.1873	0.2568	**0.3780**	0.2417
ParTiDef	0.1547	0.2256	0.3360	0.2106
PasTiDesDef	0.1631	0.2274	0.3460	0.2264

Table 3: Comparison at Opinion Relevance.

For topic relevance, ParTitDesDef yielded our best results when evaluated using MAP and P10 whereas ParTiDesDmt2 yielded best results when evaluating with Bpref and P10. Compared with the median across all 57 runs submitted by any team, ParTitDesDef outperformed the median AP for 37 out of 50 topics. For opinion relevance, ParTiDesDmt2 yielded our best results when evaluating using MAP, Bpref and P10. Compared with the median across all 57 runs, ParTiDesDmt2 outperformed the median AP for 39 out of 50 topics (see Figure 1), but hit the best AP only once. Median results are likely to be biased somewhat low in the first year of a new track, but this analysis suggests that conventional term weighting and document scoring techniques such as those implemented in Indri perform reasonably well for this task.

Demotion of opinionated documents does not help or hurt much the retrieval effectiveness at both topic relevance and opinion relevance levels. Comparing AP of ParTiDesDmt2 with ParTitDesDef topic by topic (see Figure 2) reveals that ParTiDesDmt2 does better than ParTitDesDef on some topics but worse than the others. We suspect this might be due to our flawed way of sentiment detection. Additional analysis will be needed to characterize the reasons for the ineffectiveness of our opinion detection approach.

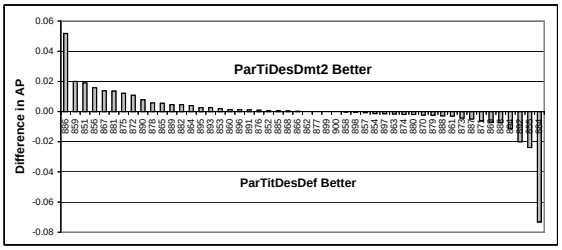

Figure 2: [Blog] Difference between ParTiDesDmt2 and ParTitDesDef, Average Precision at Opinion Relevance.

2 Enterprise Track

The primary goal of the expert search task [4] is to look for a person or multiple people in an organization who are experts on a given topic. The task briefly connects content (emails, web pages, etc.) to people and both represent the major two sides of searching (or making sense of) any informal communication media, matching the main goal of JIKD project in University of Maryland[6] that focuses more on emails as an example of such media. Hence, our first-time participation in the task has two goals. The first is to build a baseline and the second is to apply what we have learned in JIKD about modeling identity [5] to a public mailing lists such as W3C as opposed to personal archives such as Enron collection [11]. With both in mind, we have adopted a simple unsupervised approach that focuses solely on mailing lists as the source of evidence and ranks candidates based on references (in the headers and the body text) to their names and email addresses in a set of emails that the system believes as relevant to the topic of interest. The credit granted at each detected reference is computed based on (1) the relevance score of the email where the reference appears, and (2) the specific email field (headers, new text, quoted, etc.) in which it is observed.

2.1 Preprocessing

Duplicate Removal We used W3C mailing lists version that was cleaned by Daqing He[7]. Before any further processing we removed the duplicate emails, yielding 169,053 unique emails out of what was originally 173,146 emails in that cleaned version.

Detection of signature blocks Since our approach is based on the references in the body of the email, it was natural to isolate the signature blocks of each email so that the names that appear there do not mislead the subsequent scoring. To detect these blocks we used the same technique described in [5].

Indexing We indexed the mailing lists using Lucene, an open source search engine[8]. Body text was stemmed and the stop words were removed from each email and the new text, signature block, quoted headers, and quoted text were indexed in separate fields.

Building simple models of identity To be able to recognize as many references to the same person as we can, we have built simple models of identity [5] that associate multiple forms of names and email addresses of the same person from the main and quoted headers. For each person, only strong co-occurrences (whose relative frequency exceeds a specific threshold) are kept in the model in order to get rid of unreliable associations that were incorrectly extracted due to parsing errors. For the expert search task, a list of 1092 candidates (one name and one email address for each) is provided to the search system. The built models are aimed to enrich that list by alternative addresses and forms of names of the same person.

Query Formulation We have formulated our queries using one or more (concatenated) fields of the traditional TREC topic format. The specific combinations used in the submitted runs will be given later in section 2.3. Each

[6] www.jikd.umiacs.umd.edu
[7] http://www.sis.pitt.edu/~daqing/w3c-cleaned.html
[8] http://lucene.apache.org

Table 4: Fields f and Weights w_f

Sender	2.0	Receiver	1.0	Subject	1.0	New text	t_f
Qtd. sender	1.0	Qtd. receiver	0.5			Qtd. text	t_f

query was stemmed and the stop words were removed before being passed to the search engine. The searched email fields are both subject and new text.

Name Recognition We used Aho-Corasick algorithm [1] to recognize references to names or email address in linear time. Note that we restrict the list of recognized names to full names only trying to reduce the uncertainty in resolving them.

2.2 Retrieval Approach

We first retrieve a set of emails that are relevant to the formulated query using Lucene's vector space model. We experimented with two different retrieval approaches:

2.2.1 Email-based Approach

This approach considers the top 500 emails as the relevant set of emails R_T to the given topic T and computes a score for each candidate as follows:

$$score(cand|T) = \sum_{d \in R_T} support(d|cand, T)$$

where

$$support(d|cand, T) = sim(d, T) \cdot assoc(cand, d)$$
$$assoc(cand, d) = \sum_{f \in d} w_f(cand, d)$$

and $sim(d, T)$ is the similarity score assigned by Lucene for document d. The weights of the different fields are listed in Table 4.

2.2.2 Thread-based Approach

In this alternative approach, we add to R_T the first 15 emails in a breadth-first traversal of the threads H rooted by the emails retrieved by the query. Since those just added emails are not directly relevant to the topic T, we have used the following scoring function:

$$score(cand|T) = \sum_{h \in H} \sum_{d \in h} support(d|cand, T)$$

$$support(d|cand, T) = sim(root(d), T) \cdot \frac{1}{level(d, h_d)} \cdot assoc(cand, d)$$

where $level(d, h_d)$ is 1 + the distance from thread root to d.

2.3 Evaluation and Results

Each participating site was allowed to submit up to 5 runs. For each topic, the system provides a ranked list of candidates (up to 100) along with a ranked list of supporting documents (up to 20) for each candidate. The goal of the later list is to guide both the relevance judgment and evaluation processes.

Two evaluation approaches are applied to the task this year. The first (called "expert-retrieval") evaluates each run based solely on the expert ranking while the other (called "expert-support") considers a retrieved candidate relevant only if the system provides a (judged) positive support document for that candidate.

UMD submitted 5 runs (listed in Table 5) which differ in how the query is formulated given the TREC topic and whether the system retrieves emails or threads. The table gives a brief summary of the evaluation results of

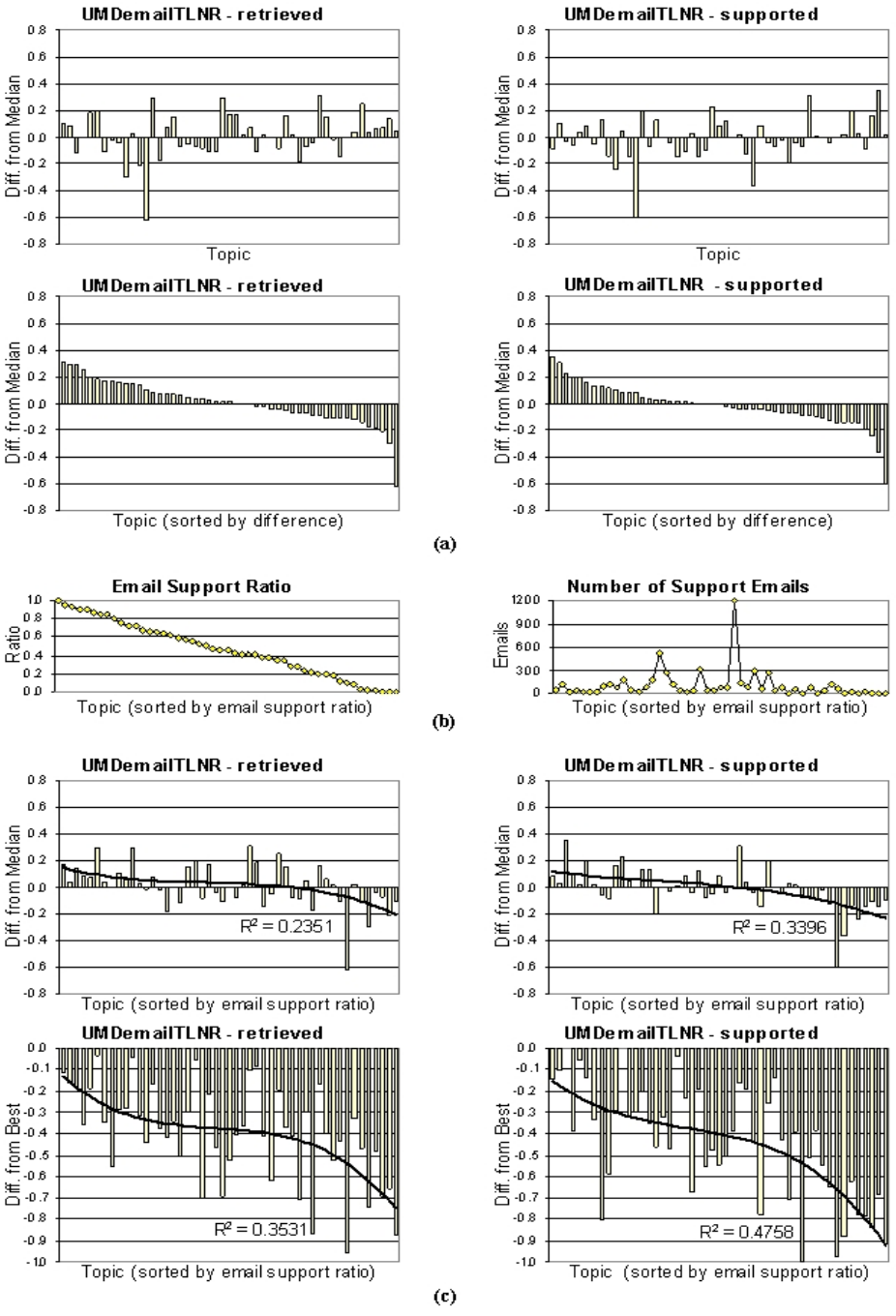

Figure 3: Expert search results per topic: (a) Difference in average precision from median, (b) Judged email support documents, and (c) Difference in average precision from median and best, sorted by ESR per topic.

Table 5: Summary of the Results

			Retrieval		**Support**	
Run Id	Query	Approach	MAP	P@10	MAP	P@10
UMDemailTTL	Title	Email	0.195	0.406	0.072	0.182
UMDemailTLNR	Title + Narr.	Email	**0.350**	*0.504*	**0.141**	**0.298**
UMDthrdTTL	Title	Thread	0.218	0.449	0.090	0.198
UMDthrdTTLNR	Title + Narr.	Thread	*0.343*	**0.514**	*0.139*	*0.294*
UMDthrdTTLDS	Title + Desc.	Thread	0.315	0.502	0.119	0.278
Average of Medians	—	—	0.341	0.508	0.154	0.294

each run in two different measures for both methods of evaluation. The average score of the median system in each topic is also listed in the table for comparison.
The table indicates that the runs which use both title and narrative parts of the topics were the best with the email-based one in particular the best in most measures. Notice the significant difference between scores in both evaluation approaches for the same run (even in the average-of-medians case), which means that systems were usually not able to justify their choices.

The results per topic for our best run (UMDemailTLNR) are shown in Figure 3(a) where the difference in average precision from the median score is plotted for each topic sorted by topic ID and that difference. The figure shows that this particular run may represent an average system among all submitted runs.

Since our approach only utilizes mailing lists part of the W3C collection, it is interesting to see how that limitation affects the performance of our approach. Here we propose one way to achieve that goal by first computing Email Support Ratio "ESR"; an estimation of the dependency of a topic T on mailing lists as follows:

$$ESR(T) = \frac{|\text{Positive Support Emails of } T|}{|\text{Positive Support Documents of } T|}$$

ESR gives an indication of how much each topic is actually represented (or discussed) in mailing lists. Figure 3(b) illustrates both ESR and the actual number of positive support emails per topic. The figure shows that ESR spans its range almost uniformly over topics but it is not directly proportional to the number of supporting emails. Two topics (EX63 and EX74) had no email support at all.

To investigate whether there is a potential relationship between our scores and ESR, we plot in Figure 3(c) the difference in average precision from both median and best scores over topics sorted by ESR. Fitted cubic curves are also plotted in each case. The curves show that our system (in average) performs better as ESR increases, i.e, scores on topics that are heavily discussed in mailing lists are generally better

3 Legal Track

In the legal community, people have been familiar with using simple term-based techniques to search against large data sets. However, the effectiveness of such technologies in finding responsive documents in legal settings has not been generally studied. The TREC legal track provides a forum to evaluate the results of searches on "topics" approximating real legal requests in litigation. For the first year of the track, the University of Maryland team mainly tried two techniques: Boolean search and combining query terms from multiple sources. Since a boolean query for each search request was provided in the test collection, we wanted to see how a retrieval system that is designed for ranked retrieval would perform if we made it act like a Boolean search system. In this case, we used Indri retrieval engine, and by combining some of its query operators we were able to produce a Boolean run. For comparative purpose, we also produce three other runs, which are explained below.

We indexed all the fields in the document collection to create one document index. All the four submitted runs used this index. The first is a baseline run, whose queries used all words from the <RequestText> field in the topics to form automatic queries. This is called baseline run since we expected other runs to achieve better retrieval effectiveness through either performing Boolean retrieval or adding query terms from the Boolean text. The second run is a run using manually formulated queries based on the <FinalQuery> field in the topics. Since we did not have a true Boolean retrieval system available, we combined several query operators in Indri retrieval engine to "mock" this Boolean run. In addition, we also submitted a run with queries based on all word from the original Boolean queries to form queries (i.e., ignored all the Boolean syntax and treated each query as a

bad of words). A comparison of this run with the above Boolean run will reveal whether the Boolean syntax can improve the retrieval effectiveness. Finally, we combined all word from the original Boolean queries and all words from the <RequestText> to form combined queries. We want to see whether using such a combination of query words can lead to improvement over using either the Boolean query words or using the <RequestText> words alone.

3.1 Techniques

Although research and practice in information retrieval has run for a long time, we haven't seen much comparison of Boolean retrieval and ranked retrieval. A main reason is that usually a retrieval system is designed for either ranked retrieval or Boolean retrieval, both rarely for both. Boolean retrieval has been widely used in the legal community, while the majority of the IR systems used in TREC evaluation are ranked retrieval. It would be very interesting to conduct such a comparison since the result can inform us whether experimental ranked retrieval systems can achieve performance comparable to or even better than the practical Boolean system. However, it is not easy for research teams like us to get a practical Boolean system. Fortunately, we found that Indri retrieval system provides several query operators that can be used to mock Boolean retrieval. So we decided to use Indri to conduct both the Boolean retrieval and the ranked retrieval. In this section, we describe how we constructed Boolean queries in Indri and other three sets of queries used in our official submission.

The retrieval model in Indri is based on a combination of the language modeling and inference network retrieval frameworks [16]. Indri provides a lot of operators that can be sued to form complicated and effective structured queries. Two types of query operators that are particular important for the purpose of this study are "matching" operator and "belief" operator. Matching operators can be used to decide whether a document match the query in question. This is basically done through some sort of term matching. Belief operators are used to decide the degree of matching. For ranked retrieval, they are directly related to the rank of a document. In our study, we used different combinations of those query operators to mimic Boolean retrieval. Specifically,

- OR For Boolean expression "a OR b", we constructed "#combine(a b)" as the Indri Boolean query. This query will return every document that contains either a or b or both. As the name indicates, #combine operator combines the belief based on term a and term b, hence ranking is possible, although for the purpose of Boolean retrieval, ranks can be ignored.

- AND For Boolean expression: "a AND b", we constructed "#filreq(#band(a b) #combine(a b))" as the Indri Boolean query. This query will return all documents containing both a and b and then rank documents according to the belief computed with "#combine(a b)". Again, the score can be ignored for Boolean retrieval.

- NOT For Boolean expression "a NOT b", we constructed "#filrej(b a)". This will score all documents that contains term a but never return any containing term b.

The above three basic operations can be combined to handle more complicated Boolean queries. For example, for a Boolean expression:

((a OR b) AND c) NOT d

We can first use Boolean logic to convert it into:

((a AND c) OR (b AND c)) NOT d

The resulting Indri query will be:

#filrej(d #combine(#filreq(#band(a c) #combine(a c)) #filreq(#band(b c) #combine(b c))))

For more complicated Boolean queries, such as "(a OR b) AND (c OR d)", #syn query operator should be used. This is because if we follow the above rules, we'll have a query like this:

#filreq(#band(#combine(a b) #combine(c d)) #combine(#combine(a b) #combine(c d)))

However, this query won't work because the #filreq (and also #filrej) operator requires the first argument to an expression that returns matches, not one that returns scores. Being a belief operator, the #combine operator returns scores, so it can't be used as the first argument. In cases like this one, the #syn operator can be used instead. #syn is a matching operator - #syn returns documents that matches at least one of the term specified in it. Consequently, the above query can be revised as:

#filreq(#band(#syn(a b) #syn(c d)) #combine(#combine(a b) #combine(c d)))

Following these rules, we were able to work around to convert the Boolean queries provided in the search topics into Indri queries. Although it is possible to automatically derive Indri Boolean queries based on the

original Boolean queries of the search topics, we used a semi-automatic approach due to some syntax errors and typos in the original queries. That is, we used a Perl script to form the initial set of Indri Boolean queries based on the above rules, and then manually corrected the errors. In the original Boolean queries, most words were truncated. In formulating the Indri Boolean queries, however, we returned to the full form of each word since we applied the built-in stemming function of Indri to both the queries and the documents.

The other three set of queries used in out official submission are for comparison purpose. For a baseline ranked retrieval run, we formulated the query by simply using all words contained in the <RequestText> field. Again, we did not perform extra operations such as removing punctuation or stemming since they were taken care of by the Indri functions. The third set of queries used all words in the original Boolean queries of the search topics, but we ignored all Boolean syntax operators. We're interested in knowing whether a ranked retrieval system that does not support Boolean retrieval can achieve reasonable retrieval effectiveness if we simply feed it with Boolean queries constructed by people who are more comfortable with creating Boolean queries. Finally, we formulated another set of queries that used a combination of all words from the <RequestText> and <FinalQuery> (i.e., the original Boolean query field). Here, we would like to see whether such a combination of query words from two sources could lead to better retrieval effectiveness than using each of them alone. Of course, these two sources share a lot of words.

As mentioned above, we used Indri retrieval engine for our experiment. Indri is retrieval system jointly developed by the University of Massachusetts and Carnegie Mellon University. It has been made freely available for the IR community [9]. Although Indri accepts documents in a variety of formats including XML and the legal documents are in XML format, it requires each document as one file. Since the collection contains 6,910,192 documents, we decided to convert the documents into trecweb format instead of splitting them into smaller files (Indri also accepts trecweb format and does not require one document per file). All the fields in the original XML format were retained as trecweb <BODY> content. Finally we create a single document index with this reformatted document collection. Porter stemmer was used in both query and document processing.

Due to time constraint, we were not able to try other variants of indexing techniques such as character n-gram. We understood that the documents contain many errors from from an optical character recognition (OCR) process, and character n-gram based retrieval may compensate some of these errors.

3.2 Results

Table 6 shows the the average precision of each topic for the four submitted runs. Overall, mean average precision (MAP) is very low for all the four runs, indicating either this is a challenging task or the techniques we explored were not very effective. We suspect that the OCR errors might be a major contributing factor, especially when word-based indexing was used.

Among the four runs, the one using all words from the <RequestText> field and the field <FinalQuery> (i.e., Comb in the table) has the highest MAP. Wilcoxon signed rank tests indicate that this run significantly outperformed the baseline run and the run with words from the Boolean queries (i.e., Base and BooleanAuto in the table, respectively), while it is indistinguishable from the Boolean run (Boolean in the table). Two things are important here. First, combing query words from different sources seemed to help improve retrieval effectiveness. Secondly, Boolean retrieval can be just as effective as the best ranked retrieval in our study. The Boolean run is statistically indistinguishable from any of the other three runs.

We were informed by the track organizers that, after the official evaluation was completed, our Boolean run was indeed evaluated as if it were a ranked retrieval.

We also looked at the number of documents returned for each topic in our Boolean run (see Column "Boolean ret#" in Table 6). In fact, the system did not return any document for eight topics (Topic numbers: 10, 17, 31, 33, 36, 38, 46, 47). However, in Table 6, each of those eight topics has one retrieved document. This is because the sanity check script used by NIST requires at least one retrieved document for each topic, so we randomly added one document for each of those eight topic in our Boolean run. Overall, the number of retrieved documents varied greatly among the topics. We're not sure what caused it, and future investigation is needed.

3.3 Future Work

Not surprisingly, for the first year of TREC legal track we could not achieve good retrieval effectiveness. We knew very little about the test collection before we started to work on it. Time constraint was another challenge to

[9] http://www.lemurproject.org/indri/

Topic	Base	Boolean	BooleanAuto	Comb	Boolean ret#	rel#
6	0.0002	0.0027	0.0001	0.0003	15	122
7	0.0086	0	0.0141	0.0284	13	165
8	0.0155	0	0.0044	0.0063	1	185
9	0.0213	0	0.071	0.0565	201	129
10	0.0001	0	0.0003	0	1	5
13	0.0253	0.061	0.0206	0.0491	208	160
14	0	0	0	0	2	36
17	0.0137	0	0.004	0.0185	1	4
18	0.0114	0.0026	0.0331	0.023	21	80
19	0.0123	0.0402	0.0289	0.0183	456	502
20	0.0003	0.0211	0.0059	0.0077	2604	35
21	0.0005	0.1393	0.0908	0.0462	218	289
22	0	0	0	0	1	68
23	0.0039	0.0051	0.0009	0.0035	9	392
24	0.0023	0	0.014	0.0353	755	9
25	0	0	0	0	2039	12
26	0.0001	0.0032	0.0001	0.0001	976	352
27	0.0011	0.008	0.0008	0.0005	1530	184
28	0.0978	0.1185	0.0084	0.1401	2647	46
29	0	0	0.0031	0.0005	12	17
30	0.2212	0.2345	0.2791	0.4789	63	93
31	0.1316	0.0031	0.1395	0.1404	1	320
32	0.0268	0.0476	0.0034	0.0362	7	63
33	0.023	0	0.0734	0.137	1	37
34	0.055	0.179	0.0447	0.0813	80	245
35	0.1119	0	0.0629	0.138	1	34
36	0	0	0	0	1	13
37	0.0046	0.0767	0.0243	0.0307	733	78
38	0.0288	0	0.0047	0.0116	1	136
39	0.0178	0.0159	0.0048	0.023	856	18
40	0	0	0	0	5	1
41	1	1	0.2	0.5	27	1
43	0.007	0	0.0032	0.003	3	161
44	0.0021	0	0	0.0009	4	28
45	0.0844	0.0064	0.0259	0.1143	2	157
46	0.0005	0.02	0.0207	0.0333	1	50
47	0.0013	0	0.0006	0.0018	1	6
50	0.0004	0	0.0448	0.0176	3	61
51	0	0	0.0001	0	299	29
MAP	0.0495	0.0509	0.0316	0.056		

Table 6: Average precision (AP) for 4 official runs. Base: queries were formulated by using all words from the <RequestText> field; Boolean: queries were Indri Boolean queries manually formulated based on the <FinalQuery> field; BooleanAuto: queries were formulated by using all words in the <FinalQuery> field (ignoring the Boolean syntax; Comb: queries were formulated by using all words from the <RequestText> field and the <FinalQuery> filed; Boolean ret#: the number of documents retrieved in the Boolean run; rel#: the number of relevant documents (official).

us. Our team had to serve as a member of the track organizing team while doing our own research. However, we gained valuable experience. Our immediate plan is to try out character n-gram based indexing and compare the results with the word-based indexing results. This may give us a better understanding of the effect of OCR errors on the retrieval effectiveness. We also would like to index selectively different fields in the document collection (e.g., metadata fields vs. OCR'd fields). In the future, we plan to team up with law librarians/students to explore a broader range of issues, such as manual query formulation based on the full complaints and interactive retrieval. Therefore, we look forward to next year's legal track!

4 QA Track

Information needs are often complex, evolving, and difficult to express or capture [17], an issue that is not well addressed in batch-oriented information retrieval system evaluations. This issue has received attention in earlier TREC's, most notably in the interactive track and in the HARD track [2]. In our previous work, we found that well-constructed clarification questions help to better characterize the user's information need and thus yield better retrieval effectiveness [14].

Interactive question answering has recently become a focus of research in the context of complex QA. The question templates in the ciQA task are substantially different from factoid questions in that the information needs cannot be answered by named entities such as people, organizations, locations, dates, etc. To investigate the role of interaction in complex QA, we relied on an approach similar to [14]: a trained intermediary manually read through relevant documents, generated answers from them, constructed clarification questions, and produced improved answers based on the clarification responses. We also submitted an automatic run to test quasi-relevance feedback for question answering. Overall, we had three major goals:

- To explore the effectiveness of single-iteration written clarification dialogs
- To explore different strategies for clarifying user needs in question answering;
- To better understand the nature of complex, template-based questions.

The description of our ciQA activities is as follows: Section 4.1 describes our manual runs. Section 4.2 describes our automatic runs. Section 4.3 presents official results.

4.1 Manual Run

This section describes our methodology for creating manual pre- and post-clarification runs (UMDM1pre, UMDM1post). We employed a trained intermediary in all phases of the process.

Document Retrieval. We employed the *building blocks* strategy [6, 15] with Lucene to obtain a set of relevant documents. We constructed a Boolean query based on the question template and narrative for each topic, augmented with various query term expansions. Concepts were ANDed, and the set of synonyms for each concept were ORed together.

We used a number of external sources for expanding query terms:

- The CIA World Factbook: used for geographic expansion. The World Factbook provides different forms of country names: conventional long form, conventional short form, local long form, local short form, and abbreviation. In addition, we also expanded country or region names to their states/provinces, ports and terminals, and major cities (all fields in the Factbook).
- Google: used for lists of instances. For example, we expanded "performance-enhancing substances" into the names of banned substances by NCAA and other organizations.
- WordNet: used to expand general concepts to their hyponyms, hypernyms, and synonyms. For example, we expanded "weapon" to its hyponyms (for example, missile) and its synonyms (for example, arm).
- Roget's Thesaurus: used for common concepts. For example, we expanded transportation to its various forms, such as barge, rail, air, water, road, cargo, freight, and so on.

- Wikipedia: used for variation of organization names and related organizations. For example, we expanded IRA to "Republican Movement", "Irish republican organizations", "Irish Republican Army", "Fianna Eireann", and so on.

The query expansion allowed us to retrieve documents at the conceptual level—beyond simple matching of terms in the topic description.

Answer Generation. We examined the top 20 documents for each topic to select statements, facts, or evidence that would answer the question. We resolved anaphora, combined multiple sentences, and eliminated redundant information when necessary.

Answer Ranking. Having gathered all the pieces, we then ranked the answers based on importance and/or logical order to generate a coherent response. The intention was to present the answers in a order that makes sense to the assessor. Depending on the topic, this could be chronological order, order of importance, or a summary statement followed by supporting evidence.

Clarification Forms Generation. We used three types of clarification questions, depending on how well we understood the question and the answers after the initial run.

- For 10 questions where the topic narrative seemed insufficient to generate a coherent response, we asked specifically for guidance. See Figure 4 for an example.
- For 12 questions where the topic narrative seemed clear and the answers seemed to roughly cluster into categories, we asked the assessor about the importance of the various categories so that we can re-rank the answers. See Figure 5 for an example.

  In this form, we also presented a few sample answers. Assessors were asked to judge the relevance of each sample answer; the choices were relevant, partially relevant, and not relevant.
- For 8 questions where answers did not seem to cluster into coherent categories, we picked a number of sample answers for soliciting relevance feedback. The choices were relevant, partially relevant, and not relevant.

Refinement of Final Answers. After we received feedback from the users, we reselected and reranked the initial answers. Additional searches were performed for a few topics. The number of manual answers for each topic ranged from 5 to 18. We then filled the response with automatically-selected sentences retrieved from the manual queries until the 7,000 character limit was reached.

4.2 Automatic Run

Our process for generating the automatic runs (UMDA1pre, UMDA1post) was relatively simple. For each topic, the question and narrative was used verbatim as a query to Lucene. From the top 20 resulting documents, 10 terms were selected based on *tf.idf* values. These terms were added to the original query to retrieve a new set of documents. Sentences in this set that did not contain at least one term from the question were discarded. The remaining sentences comprised the initial run.

The automatically-generated clarification forms asked the assessor for relevance judgments on each of the sentences (relevant, partially relevant, not relevant, and don't know). The final run was prepared by moving sentences judged as relevant up to the top of the list, followed by the sentences judged partially relevant. Sentences judged as not relevant were discarded.

4.3 Results

Official NIST results are shown in Table 7. For our automatic run, interaction actually decreased performance substantially. We have noticed that there exists a non-straightforward connection between sentence-level relevance judgments and the "nuggetization" process involved in the evaluation methodology—these and related issues are currently under investigation.

Topic 26: What evidence is there for transport of [smuggled VCDs] from [Hong Kong] to [China]?

1. What types of smuggled disks are you interested in? Check all that apply:
   - ☐ VCDs
   - ☐ CDs
   - ☐ DVDs
   - ☐ Other. Please specify:
2. Please rank the following types of evidence in order of importance to you (as 1, 2, 3..., with 1 the most important). If two or more types are of the same importance, put the same number for them.
   - ☐ Volume of smuggled VCDs, for example: The Customs police in south China's Guangdong Province recently confiscated more than 210,000 pirated video compact disks smuggled into the province from Hong Kong.
   - ☐ Ruses used by smugglers, for example: Smugglers hid these CDs and VCDs inside waste papers carried by a ship coming from Hong Kong and transferred them to a small wooden boat.
   - ☐ Other. Please describe:

Figure 4: Sample topic and clarification form.

Topic 26: What effect does [aspirin] have on [coronary heart disease]?
Please rate the importance of following types of evidence.

1. General claim of effects of aspirin.
   ☐ Important. ☐ Somewhat important. ☐ Not needed at all.
2. Guideline of how aspirin can be used to treat heart diseases.
   ☐ Important. ☐ Somewhat important. ☐ Not needed at all.
3. How aspirin works to prevent heart diseases.
   ☐ Important. ☐ Somewhat important. ☐ Not needed at all.
4. Side effects of aspirin.
   ☐ Important. ☐ Somewhat important. ☐ Not needed at all.
5. Facts that people take aspirin to treat certain heart diseases.
   ☐ Important. ☐ Somewhat important. ☐ Not needed at all.
6. Claims of doctors or medical researchers.
   ☐ Important. ☐ Somewhat important. ☐ Not needed at all.
7. Other. Please describe:
   ☐ Important. ☐ Somewhat important. ☐ Not needed at all.

Figure 5: Sample topic and clarification form.

Run	**F-Score**
UMDM1pre	0.316
UMDM1post	0.350 (+10.6%)
UMDA1pre	0.224
UMDA1post	0.180 (−19.4%)

Table 7: Official results for the ciQA task.

Acknowledgments

Thanks to Trevor Strohman and Don Metzler for their help with Indri. This work was supported in part by the the DARPA GALE program and by the Joint Institute for Knowledge Discovery.

References

[1] A. V. Aho and M. J. Corasick. Efficient string matching: an aid to bibliographic search. In *Communications of the ACM*, 1975.

[2] James Allan. HARD track overview in TREC 2005: High accuracy retrieval from documents. In *Proceedings of the Fourteenth Text REtrieval Conference (TREC 2005)*, 2005.

[3] Blood, Rebecca, 2002. *The weblog handbook: practical advice on creating and maintaining your blog.* Perseus Publishing, Cambridge, MA.

[4] P. De Vries A. Craswell, N. and I. Soboroff. Overview of the TREC-2005 enterprise track. In *Working Notes of 2005 Text Retrieval Conference (TREC 05)*, 2005.

[5] T. Elsayed and D. W. Oard. Modeling identity in archival collections of email: A preliminary study. In *Conference on Email and Anti-Spam*, pages 95–103, Mountain View, California, July 2006.

[6] Stephen Harter. *Online Information Retrieval: Concepts, Principles, and Techniques.* Academic Press, San Diego, California, 1986.

[7] Hatzivassiloglou, Vasileois, and McKeown, Katheleen, 1997. Predicting the semantic orientation of adjectives. *ACL-97*, 1997, 174-181

[8] Hearst, Marti, 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23 (1), pp.33-64, March 1997.

[9] Herring, Susan C., Scheidt, Lois Ann, Wright, Elijah, and Bonus, Sabrina, 2004. Bridging the Gap: A Genre Analysis of Weblogs. *Proceedings of the 37th Hawaii International Conference on System Sciences*, 2004

[10] Kennedy, Alistair and Inkpen, Diana, 2005. Sentiment classification of movie and product reviews using contextual valence shifters. *FINEXIN 2005.*

[11] B. Klimt and Y. Yang. Introducing the Enron corpus. In *Conference on Email and Anti-Spam*, Mountain view, CA, USA, July 30-31 2004.

[12] Lasica, J. D., 2001. Blogging as a form of journalism. *USC Annenberg Online Journalism Review.*

[13] Lasica, J. D., 2001. Weblogs: A new source of news. *USC Annenberg Online Journalism Review.*

[14] Jimmy Lin, Philip Wu, Dina Demner-Fushman, and Eileen Abels. Exploring the limits of single-iteration clarification dialogs. In *SIGIR 2006*, pages 469–476, 2006.

[15] Gary Marchionini. *Information Seeking in Electronic Environments.* Cambridge University Press, Cambridge, England, 1995.

[16] D. Metzler and W.B. Croft. Combining the language model and inference network approaches to retrieval. In *Information Processing and Management Special Issue on Bayesian Networks and Information Retrieval*, volume 40, pages 735–750, 2004.

[17] Robert S. Taylor. The process of asking questions. *American Documentation*, 13(4):391–396, 1962.

[18] Turney, Peter and Littman, Michael, 2003. Measuring praise and criticism: inference of semantic orientation from association. *ACM Transactions on Information Systems (TOIS)*, 21. 315-346.

[19] Wiebe, Janyce, Wilson, Theresa, Bruce, Rebecca, et al, 2002. Learning Subjective Language. Tech Report TR-02-100, Dept. of Comp. Science, Univ. of Pittsburg.

[20] Wilson, Theresa, Wiebe, Janyce, and Hoffmann, Paul, 2005. Recognizing contextual polarity in phrase-level sentiment analysis. *HLT-EMNLP 2005.*