Structured Queries for Legal Search

Yangbo Zhu Le Zhao Jamie Callan Jaime Carbonell

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{ yangboz, lezhao, callan, jgc }@cs.cmu.edu

ABSTRACT

This paper reports the experiments of using Indri for the main and routing (relevance feedback) tasks in the TREC 2007 Legal Track. For the main task, we analyze ranking algorithms using different fields, boolean constraints and structured operators. Evaluation results show that structured queries outperform bag-of-words ones. Boolean constraints improve both precision and recall. For the routing task, we train a linear SVM classifier for each topic. Terms with the largest weights are selected to form new queries. Both keywords and simple structured features (term.field) have been investigated. Named-Entity tags, LingPipe sentence breaker and metadata fields of the original documents are used to generate the field information. Results show that structured features and weighted queries improves retrieval, but only marginally. We also show which structures are more useful. It turns out metadata fields are not as important as we thought.

1. INTRODUCTION

The goal of legal search is to retrieve all relevant documents for production requests. A production request describes a set of documents that the plaintiff forces the defendant to produce. The plaintiff usually forms comprehensive requests to cover large amount of documents that are potentially useful in the trial. Due to the high risk of missing important documents, legal search systems are usually recall-oriented.

Legal Track 2006 and 2007 both use the IIT CDIP collection [1]. It consists of 6,910,192 business records from US tobacco companies and research institutes. Each document contains OCR text and multiple fields of metadata, including title, authors, organizations, etc.

For the main task, 50 topics from Legal 2006 are used as training data. For evaluation, 50 new topics are generated from four hypothetical complaints. Each topic contains detailed information about the background, instruction and negotiation history. Four fields are especially important for retrieval. First, the plaintiff describes the desired documents in *RequestText* (RT). Based on RT, the defendant proposes a boolean query called *ProposalByDefendant* (PD). Then the plaintiff modifies PD to form a new query *RejoinderByPlaintiff* (RP). Finally, the two parties agree on a *FinalQuery* (FQ), which is actually used to retrieve documents.

For the routing task, 10 topics from 2006 are adopted for evaluation. Systems take advantage of existing relevance judgements to retrieve more relevant documents. During evaluation, all previously judged documents are filtered out from the ranked lists, and performance metrics are calculated based on newly judged documents.

The rest of this paper is structured as follows. Section 2 and 3 describe our methods and experiments for the main and routing tasks. We conclude with section 4.

2. MAIN TASK

This section introduces our experiments for the main task. First we describe the formation of Indri queries based on request topics. Then we compare the results of different runs using various boolean constraints and ranking functions.

2.1 Query Formulation

A typical Indri query in our experiments has two components: the boolean constraint and the ranking algorithm. For each topic, we convert *FinalQuery* to a boolean constraint, and combine terms from different fields for ranking. When the original query contains 'BUT NOT', the Indri query is of the general form

#filrej(Z #filreq(#band(X) #combine(Y)))

Otherwise, it's of the simpler form

#filreq(#band(X) #combine(Y))

Here, X and Z are boolean constraints, and Y is the ranking component.

[1]http://www.ir.iit.edu/projects/CDIP.html

2.1.1 Wildcards

Most queries in the data set contain lots of wildcards. For example, "boost!" matches all words with prefix "boost". Since the OCR texts contain errors, there are typically thousands of matches to one wildcard expression. If we directly translate these wildcards to Indri wildcards, the query execution will be very time consuming.

To speed up the retrieval, we expand the wildcards as query pre-processing. From all the terms matching a wildcard, top K words with the highest document frequencies are selected. For instance, given wildcard "multipl!", "multiple", "multiply" and "multiplicity" are selected. This method dramatically reduces the number of inverted lists to be merged at query time. Since top K frequent words cover the majority of matches, the potential loss in recall is low. However, queries expanded by this method does not perform well. One reason is that wildcards bring in noisy words. For example, in topic 52, "high!" is expanded to "highlight", "highway" and "highland". In topic 81, "bee!" is expanded to "beer", "beef", "beach", etc.

By examining the original queries, we find that most of the wildcards are unnecessary if we use stemmer during indexing. In this paper, wildcards are manually expanded to one or multiple lexically-related terms according to the topic description. A special case is for years: "198!" is expanded to "#syn(1980, 1981, ..., 1989)".

2.1.2 Boolean Constraints

Each topic in the Legal Track provides three boolean queries: *ProposalByDefendant*, *RejoinderByPlaintiff* and *FinalQuery*. They contain boolean and proximity operators. We build a parser to parse original queries into trees, and then convert the trees to Indri queries. Table 1 shows the basic mapping rules, similar to those used in [2]. For example, an original query "((a OR "b c") AND d BUT NOT e)" is converted to Indri query #filrej(e #filreq(#band(#syn(a #1(b c)) d) #combine(. . .))).

Table 1: Mapping from original operators to Indri operators

Original expression	Indri expression
"x y"	#1(x y)
x W/k y	#uw(k+2)(x y)
x OR y	#syn(x y)
x AND y (inner)	#uw(x y)
x AND y (outermost)	#band(x y)
x BUT NOT z	#filrej(z x)

2.1.3 Ranking

Suppose the original query is "((a OR "b c") AND d)". The corresponding bag-of-words ranking component would be "#combine(a b c d)".

If the phrase operators are respected, the ranking component becomes "#combine(a #1(b c) d)". Since a phrase usually has much higher IDF than any of its composing terms, the phrase has high impact on ranking.

If we view terms connected by 'OR' as synonyms, the ranking component becomes "#combine(#syn(a #1(b c)) d)". For example, in topic 6, "TV", "television" and "cable" are synonyms. They may have different document frequencies, but they are treated as the same word after applying the #syn operator. The TF of a synonym set is the sum of TF of all its members, and the DF of a synonym set is the number of documents containing any of its members.

2.2 Experiments

We use the program kindly provided by Howard Turtle to transform original XML files to TREC Web format. During preprocessing, word segments separated by hyphens are connected [2]. We use Porter Stemmer during indexing. Since we don't use wildcards in Indri queries, stemming compensate loss in recall to some extent.

Scanned documents contains OCR errors. If the error rate is high, it is worthwhile to correct errors before indexing. Observation shows that OCR errors in the CDIP collection are almost character-wise "context-free", which means the (mis)recognition of a character does not depend on the characters around it. Experiments show that per character error rate is around 1%, which is acceptable. We don't correct OCR errors in this paper.

2.2.1 Submitted runs

Nine runs are produced for 2007 queries, eight of them are submitted for official pooling and evaluation. 25000 results are produced for each query.

CMUL07STD is the standard condition run required by Legal Track. It takes the keywords in *RequestText* to form a bag-of-words query using the "#combine" operator in Indri. Common query headers (e.g. "Please produce any and all documents that discuss") that are not meaningful to the topic are removed.

CMUL07O1 is an Okapi ranked list using terms in *FinalQuery*. The parameters of BM25 function are the same with those used in [4]: $k_1 = 1.2$, $k_2 = 0$, $k_3 = 8$ and $b = 0.75$. CMUL07O3 is the same with CMUL07O1 except that it combines terms from three fields: *ProposalByDefendant*, *RejoinderByPlaintiff* and *FinalQuery*.

CMUL07IRT is the bag-of-words query using keywords from *FinalQuery*. It ignores boolean constraints. CMUL07IBT is the same with CMUL07IRT except that it uses boolean constraints to filter ranked lists. If the filtered list has less than 25000 results, top ranked results from CMUL07IRT are appended to the end of the list. Duplicate documents are removed.

CMUL07IRP is the same with CMUL07IRT except that it respects phrase operators in ranking. This run is not submitted because each group can submit up to eight runs. CMUL07IBP is the same with CMUL07IRP except that it uses boolean constraints. It appends list with results from CMUL07IRP.

[2]On average, each document contains 5 hyphens at the end of lines.

CMUL07IRS is the same with CMUL07IRP except that it treats all terms connected by "OR" as synonyms. CMUL07IBS is the same with CMUL07IRS except that it uses boolean constraints. It appends list with results from CMUL07IRS.

2.2.2 *Evaluation Results*

The legal community is more interested in recall than precision. Legal 2007 takes a novel sampling method (the L07 method) to support deep pooling. Systems are required to return 25000 documents for each query. The sampling probability of a document is inversely proportional to its highest rank in all submitted runs.

Table 2 shows the evaluation results on 43 topics from Legal 2007. **RefL07B** is a reference boolean run provided by the organizers. **Median** is the median value over all 70 submitted runs, and **Max** is the maximum value. The reference run strictly follows the *FinalQuery* and provides a B value for each query. The B value is the number of documents matching *FinalQuery*. According to the sampling method, the organizers recommend **estRB** as the primary measure, which is the estimated Recall@B. We use **estPB** (estimated precision at B) as an auxiliary metric.

JudgedB is the judged documents at B. Okapi based runs have much more documents judged than Indri based ones. We suspect it is because most participating groups are using Okapi ranking, as the case in last year. If there are large amount of similar runs, the L07 method tends to sample more documents from those runs.

As last year, the reference run still outperforms all submitted runs. Among our nine runs, IBS is the best performing run in terms of both estRB and estPB. Comparing O1 and O3, using three fields does help. Since the *FinalQuery* usually covers all terms mentioned in *ProposalByDefendant* and *RejoinderByPlaintiff*, the improvement primarily comes from better term weighting. Comparing IBT and IRT, boolean filters significantly improve performance. Comparing IBT and IBP, using phrase operators alone actually hurts performance a little bit. Comparing IBT and IBS, synonym operators improve both precision and recall.

Table 2: Performance on 43 topics of Legal 2007, with estRB as the primary measure. (*IRP is not submitted)

Run	judgedB	**estRB**	estPB	estR25K
RefL07B	108	0.216	0.292	–
Max	158	0.216	0.292	0.470
Median	122	0.132	0.207	0.317
STD	138	0.123	0.191	0.314
O1	145	0.152	0.204	0.361
O3	**152**	0.170	0.236	**0.400**
IRT	136	0.132	0.189	0.295
IRP*	130	0.138	0.188	0.291
IRS	126	0.194	0.242	0.395
IBT	118	0.187	0.261	0.391
IBP	114	0.183	0.252	0.392
IBS	117	**0.208**	**0.267**	0.392

Table 3 compares different methods on 2006 and 2007 topics. Since Legal 2006 evaluation method does not support estimated metrics, we use traditional metrics. Okapi ranking using three fields achieves the highest R@B and MAP. Since the sampling methods adopted by Legal 2006 and 2007 are vastly different [1], the comparison should be taken with a grain of salt.

Table 3: Performance in Legal 2006 and 2007, using traditional metrics. (*IRP is not submitted)

	2006		2007	
Method	R@B	MAP	R@B	MAP
RefL07B	0.525	–	0.486	–
Max	–	–	0.609	0.172
Median	–	–	0.452	0.092
STD	0.469	0.084	0.485	0.115
O1	0.512	0.083	0.524	0.127
O3	**0.580**	**0.098**	**0.568**	**0.142**
IRT	0.487	0.074	0.480	0.115
IRP*	0.474	0.068	0.468	0.110
IRS	0.520	0.087	0.510	0.094
IBT	0.532	0.087	0.506	0.128
IBP	0.525	0.082	0.489	0.127
IBS	0.488	0.091	0.504	0.108

Figure 1 compares per-topic estRB between CMUL07IBS and median performance of 39 manual runs from all the groups. 30 out of 43 queries performs better than median, and four of them (60, 71, 84 and 97) achieve the highest estRB among all runs. Figure 2 compares per-topic estPB. 30 queries are above the median, and two of them (60 and 96) achieve the highest estPB.

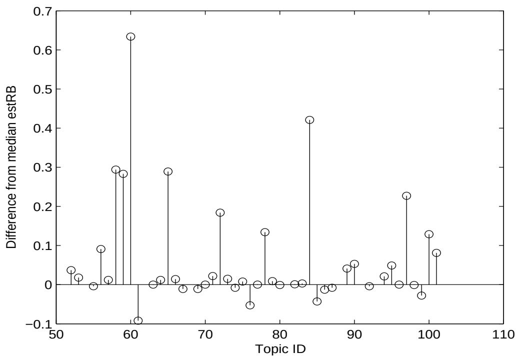

Figure 1: Main task: difference from median estRB of 39 manual runs

2.2.3 *Error Analysis*

Table 4 lists six topics on which RefL07B and CMUL07IBS behave most differently. There are three major reasons for the performance gap: chained proximity, wildcards and estimation error.

We manually expand wildcards with relevant words, and use Porter stemmer to aggregate words with the same root. This

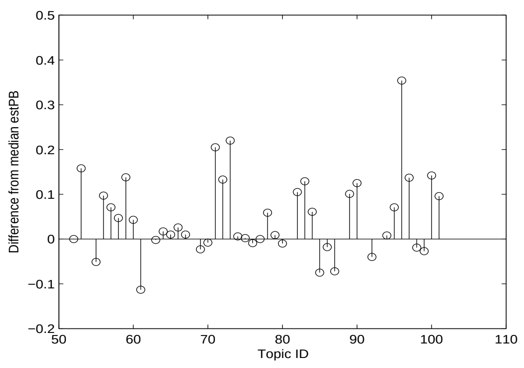

Figure 2: Main task: difference from median estPB of 39 manual runs

strategy works for most of the queries. However, when wildcards match unexpected words, it may either improve or hurt performance. Here we show an example where wildcards luckily improve estRB. Topic 72 requests "all documents referring to the scientific or chemical process(es) which result in onions have the effect of making persons cry". The *FinalQuery* is:

((scien! OR research! OR chemical) w/25 onion!) AND (cries OR cry! OR tear!)

Intuitively, we expand "cry!" to "cry". Comparing the results of RefL07B and CMUL07IBS, the latter misses two relevant documents: "gar43d00" and "brq10e00", neither of which contains any form of "cry" or "tear". However, they contain non-relevant terms such as "crystalline" and "cryptococcus", and hit the relevant documents.

Some final queries contain the "chained proximity" operators: "x W/k_1 y W/k_2 z", which requires the same occurrence of word "y" to satisfy "x W/k_1 y" and "y W/k_2 z". One possible approximation in Indri is

$$\#\text{band}(\#\text{uw}(k_1+2)(\text{x y}) \quad \#\text{uw}(k_2+2)(\text{y z}) \quad \#\text{uw}(k_1+k_2+3)(\text{x y z}))$$

However, the "y" in the Indri expressions could be different occurrences of the same word. Therefore, the Indri expression relaxes the original constraint. For topics 58 and 61, the relaxation hurts estRB, but for topic 57 and 59, it improves estRB.

In the L07 evaluation method [3], the number of relevant documents is estimated as

$$\text{estR} = \sum_{i=1}^{n} \frac{1}{p_i}$$

where n is number of judged relevant documents, and p_i is the sampling probability of document i. Although the estimator is unbiased, it has high variance when n is small. It is often dominated by relevant documents with low sampling probability. For topic 60, six documents are judged as relevant for both RefL07B and CMUL07IBS at B (1496). Between the two sets of relevant documents, only one is different. RefL07B gets "chg09d00" ($p = 1.0$), while CMUL07IBS gets "ake51c00" ($p = 0.02$). The latter gets much higher estRB because of a single document with low sampling probability. For topic 75, the number of judged relevant documents for RefL07B and CMUL07IBS at B (788) are three and eleven, respectively. One of the three documents by RefL07B ("twh67c00") has $p = 0.03$, while all eleven documents by CMUL07IBS has $p = 1.0$. Consequently, RefL07B achieves much higher estRB.

Table 4: Topics on which RefL07B and CMUL07IBS perform most differently (measured in estRB)

Topics	58	59	60	61	65	72
RefL07B	0.940	0.009	0.072	0.439	0.672	0.779
IBS	0.424	0.410	0.706	0.048	0.962	0.304
Diff	-0.516	0.401	0.634	-0.391	0.290	-0.475

3. ROUTING TASK

In the routing task, 39 topics from last year's (2006) main task with their judgments were used to simulate the routing task scenario where the system is given the information need and judgments of some of the documents returned from an initial retrieval. The routing task could be seen as a relevance feedback task, as true relevance information is known for some documents. In the experiments, at training stage, two thirds of the judged documents from 2006 were used as training while one third were used as validation data. For all 39 topics with judgments, 2 of them have only 2 judged relevant documents which were backed off to the original queries, instead of the feedback queries.

3.1 SVM Based Feedback

Given enough training documents, the relevance feedback task could be formulated as a supervised document classification problem, disregarding the original query completely. In fact, each query has on average over 100 positive training examples within over 800 total training samples. Thus, treating the retrieval for each query as a binary classification task allows us to apply state of the art classification algorithms to solve the problem. As long as the document collection does not change, discarding the original query and using only the training documents would still give reliable results.

3.1.1 Feature Selection for Classification

In text classification, it is well known that feature selection improves classification accuracy [5]. We try to investigate whether feature selection helps in this noisy OCR corpus for improving precision and recall. In our experiments, top 400 terms that have the highest correlations with relevance are the selected features for classification. The correlation is measured by Information Gain as in [5].

3.1.2 Term Selection & Expansion

Next, a linear SVM classifier [3] with all default parameters is trained to obtain a linear classification model, which is just a weight vector for linear combination of feature values. The final list of words/features that corresponds to the highest SVM weights are selected and used to construct a new query.

[3]http://svmlight.joachims.org/

3.1.3 *SVM Classification using Indri Queries*

Since the SVM term selection and feedback retrieval is just trying to use a query to approximate an SVM, we could approximate even better than just un-weighted keyword queries. When plugging in the weights learnt with SVM into the #weight operator of Indri query language [4], when the feature values of the training samples are just TFIDF scores similar to that of the retrieval model, the resulting weighted term query will be effectively classifying and ranking each document according to the linear SVM classifier.

Effectively, the retrieval system is using the inverted index to help classify and rank documents. How is this possible? Linear SVM classifier based on TFIDF features is:

$$\text{Score}_{\text{SVM}}(D|\vec{w}) = \sum_{t_i \in V} w_i * \text{tfidf}(t_i, D)$$

where the w's are SVM weights learnt from training data, V is the set of all terms in the collection. The evaluation of a weighted sum operator of the Indri query language on the document D is just

$$\text{Score}_{\text{LM}}(q, D) = \sum_{t_i \in q} w_i * \log P(t_i|D)$$

$$\text{where} \quad q = \#\text{weight}(w_1 t_1, .., w_N t_N)$$

As long as the term feature values are similar during training and at retrieval time, the weights learnt from a SVM model should be helpful to other similar retrieval models as well. In the experiments, for the term feature values at training time, we used

$$\frac{tf_{t,D} * (1 + k_1)}{tf_{t,D} + k_1 * (1 - b + b * len(D)/\text{avgLen})} * log(\frac{N}{df_t})$$

which is a variant of the Okapi BM25 formula with a modified *df* component so that there are never negative weights. At retrieval time, the weighted retrieval in Indri uses Dirichlet smoothing to generate term tfidf scores.

3.2 Simple Structured Features

Besides keywords, words that appeared in a particular metadata field or Named-Entity (NE) annotation could also be used to enlarge the feature space, so that the classifier could pick up features that predict relevance better than simple keywords. This new term consisting of the term + its field information can also be used in document retrieval (in Indri query language, simply "term.field", for example, "bush.person" where person is a named-entity tag which helps disambiguate "bush"). In the experiments BBN's Named-Entity tagger – Identifinder and the LingPipe [5] sentence breaker are used for generating additional annotations. The metadata fields are also included in generating term.field features.

One interesting thing to know is whether annotations would help retrieval. We investigate whether and how many of these structured features have been weighted highly by the SVM learner (linear SVM).

[4]http://www.lemurproject.org/
[5]http://www.alias-i.com/lingpipe/

3.3 Experiments

For different feedback retrieval algorithms, we report results comparing SVM term selection & expansion method (denoted as ex), SVM approximation using weighted query with only positive weight features (wq) and weighted query of positive and negative weight features (wqn). For different feature sets, we compare unstructured keyword features (kw) with structured term.field features (f). For unweighted term query (ex) we used top 40 high weight terms from SVM model. For wq and wqn runs, we used top 70 positive weight terms and top 100 terms with highest absolute weights respectively. These parameters are trained on the judged documents from Legal 2006. As the vocabulary of the OCR collection is noisy and huge, words with DF smaller than 50 have been discarded. Also, because some of the training documents are huge, only the first 40,000 "term.field" features have been included for each document and further reduced to 400 such structured features that are highly correlated with relevance for each query, shared among the training documents.

To compare the effects of different evaluation metrics, MAP, RecallB and est_RB are used in evaluation. Before evaluating these routing runs, all documents used in creating feedback queries, are excluded from the final result set. These documents are the judged relevant and non-relevant documents from TREC Legal Track 2006. The newly assessed ones have been used to evaluate the effectiveness of the routing task methods.

Table 5 shows the evaluation scores for the different methods as evaluated on the TREC 2006 Legal Track assessments. In TREC Legal Track 2006, 39 topics have been evaluated, and on average each topic has over 100 judged relevant documents within about 800 total judged documents. We randomly splitted the data and used 2/3 as training, 1/3 for evaluation. Results show that 1) same as in the main task, using boolean filter helps improve retrieval effectiveness, 2) using weight is better than unweighted term expansion and 3) although lots of the expansion terms are structured (see Table 7), the increase in retrieval effectiveness is only marginal — maybe these "term.field" structures are not complex/precise enough to be more accurate than keywords as indications of relevance.

Table 5: Performance on 39 topics of Legal 2006 Routing task, with 2/3 documents for training, 1/3 for validation. kw: keyword feature only, f: NE+sentence+metadata field features, f w/o meta: exclude metadata fields.

	measures	ex	wq	wqn
kw	MAP	0.1275	0.1396	**0.1408**
	Recall@B	0.1990	0.2013	**0.2034**
	R-Prec	0.1658	**0.1686**	0.1684
f	MAP	0.1427	**0.1480**	0.1469
	Recall@B	0.1976	0.2001	**0.2011**
	R-Prec	0.1592	0.1663	**0.1690**
f w/o meta	MAP	0.1423	0.1458	**0.1464**
	Recall@B	0.1976	0.2002	**0.2011**
	R-Prec	0.1623	0.1692	**0.1703**

Table 6 evaluates the routing task methods on the newly judged 10 topics out of all 39 topics of Legal 2006.

Table 6: Performance on 10 topics of Legal 2007 Routing task. Topics and feedback documents are from Legal Track 2006. (bf: short for boolean filter. RBase is run CMU07RBase which is the boolean query run, SVME is the term expansion run CMU07RFBSVME, SVMNP is the weighted query run CMU07RSVMNP which includes negative weight keywords also)

	measures	ex	wq	wqn
kw	MAP	0.0853	**0.0901**	0.0852
	Recall@B	0.5736	0.5752	**0.5765**
	est_RB	**0.3614**	0.3471	0.3476
f	MAP	0.0644	**0.0722**	0.0548
	Recall@B	0.5738	**0.5766**	0.5667
	est_RB	**0.3679**	0.3601	0.3617
f w/o meta	MAP	**0.0637**	0.0605	0.0454
	Recall@B	0.5738	**0.5766**	0.5659
	est_RB	**0.3679**	0.3601	0.3616
	submitted runs	RBase	SVME	SVMNP
	MAP	0.0976	0.1248	**0.1386**
	Recall@B	0.5691	**0.5779**	0.5038
	est_RB	0.3530	0.3342	**0.3803**

As seen from Table 6, the est_RB measure is more correlated with MAP than with Recall@B. Consistent with the results on the development set, structured features and weighted queries help retrieval a bit, but not significant.

Table 7: Percentage of structured features in the top 100 features selected out by SVM, averaged over 37 topics where there are enough training documents. The rest are keywords which constitute less than half of the feedback terms.

	NE	sen	meta
percentage	26.56%	26.24%	5.291%

In table 7, we summarize on average the percentage of all types of features being selected within top 100 as given by the weights from SVM. Although "term.field" structured features constitute slightly more than half of the high weight terms, the increase in effectiveness of the structured feedback runs are only marginally better than keywords only. More accurate structures are yet to be found. Contradictory to our intuition, according to both table 5 and 6, the metadata fields only increased precision (MAP) a bit, but had no effect or even decreased recall (Recall@B or est_RB). Maybe for human lawyers, it is simpler and more maintainable to use the metadata field in helping retrieval, but for the routing task, machine can do better with much more field information from Named-Entity fields etc..

4. CONCLUSIONS

This paper reports our experiments using Indri structured queries to retrieve legal documents in TREC Legal 2007. Legal search is special in that it is more concerned with recall at deep cut-off point. This is because lawyers usually go through thousands of documents. Finding or missing an important document may have high impact on the result of the trial.

In the main task, we compare runs with or without boolean constraints, and runs using different fields of legal requests. We study the impact of phrase and proximity operators. We treat "OR" connected words as synonyms. Experimental results show that imposing boolean constraints improves both precision and recall. Combining multiple fields gets better term weights. Structured queries significantly outperform bag-of-words ones.

In the routing task, as compared to the baseline of simple queries of combined keywords, weighted term queries and simple structured queries help retrieval only marginally. Also, more than half of the SVM selected terms are structured. Named-Entity and sentence fields appear far more often in the SVM high weight features than do metadata fields. Given the performance, more accurate structured features need to be designed in order to show a more significant improvement over simple keyword feedback queries.

5. ACKNOWLEDGEMENTS

This research was supported in part by National Science Foundation grant IIS-0707801, IIS-0534345 and CCR-0122581. Any opinions, findings, conclusions, or recommendations are the authors' and do not necessarily reflect those of the sponsor.

6. REFERENCES

[1] Jason Baron, David Lewis, and Douglas Oard. TREC 2006 Legal Track Overview. In *Proceedings of the 15th Text Retrieval Conference*, 2006.

[2] Douglas Oard, Tamer Elsayed, Jianqiang Wang, and Yejun Wu. TREC 2006 at Maryland: Blog, Enterprise, Legal and QA Tracks. In *Proceedings of the 15th Text Retrieval Conference*, 2006.

[3] Stephen Tomlinson, Douglas Oard, Jason Baron, and Paul Thompson. Overview of the TREC 2007 Legal Track. In *Proceedings of the 16th Text Retrieval Conference*, 2007.

[4] Miao Wen and Xiangji Huang. York University at TREC 2006: Legal Track. In *Proceedings of the 15th Text Retrieval Conference*, 2006.

[5] Yiming Yang and Jan Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of ICML 1997, 14th International Conference on Machine Learning*, 1997.