UMass Amherst and UT Austin @
The TREC 2009 Relevance Feedback Track

Marc-Allen Cartright and Jangwon Seo
Center for Intelligent Information Retrieval
University of Massachusetts, Amherst
{irmarc,jangwon}@cs.umass.edu

Matthew Lease
School of Information
University of Texas at Austin
mlease@austin.utexas.edu

Abstract

We present a new supervised method for estimating term-based retrieval models and apply it to weight expansion terms from relevance feedback. While previous work on supervised feedback [Cao et al., 2008] demonstrated significantly improved retrieval accuracy over standard unsupervised approaches [Lavrenko and Croft, 2001, Zhai and Lafferty, 2001], feedback terms were assumed to be independent in order to reduce training time. In contrast, we adapt the AdaRank learning algorithm [Xu and Li, 2007] to simultaneously estimate parameterization of all feedback terms. While not evaluated here, the method can be more generally applied for joint estimation of both query and feedback terms. To apply our method to a large web collection, we also investigate use of sampling to reduce feature extraction time while maintaining robust learning.

1 Introduction

Term-based models have a long and distinguished history in information retrieval, spanning vector-space [Salton and Buckley, 1987], probabilistic [Sparck Jones et al., 2000], and language modeling approaches [Ponte and Croft, 1998]. While such models are remabarkably expressive in the range of possible document rankings they can represent, their practical accuracy depends heavily on effective estimation. A wide variety of different term weighting schemes have been proposed over the years based on hand-tuning [Salton and Buckley, 1987], unsupervised learning [Mei et al., 2007], and supervised learning [Fuhr and Buckley, 1991, Bendersky and Croft, 2008, Lease et al., 2009, Kumaran and Carvalho, 2009]. While previous work in query expansion has traditionally focused on unsupervised approaches [Lavrenko and Croft, 2001, Zhai and Lafferty, 2001], recent work in supervised learning has also invesigated this scenario and shown that supervision can be beneficially applied here as well [Cao et al., 2008]. We describe a new such approach for supervised learning of expansion term weights.

Previous work in estimating term weights has generally relied on simplifying independence assumptions in order to achieve more tractable training. For example, Bendersky and Croft [2008] predict a "key concept" for each query and produce a weighted combination using classifier confidence of each independent prediction. Cao et al. [2008] similarly predict "good" terms and create a weighted combination in the same way. While Lease et al. [2009] leverage full parameterizations in estimating expected term weights (i.e. their training data), term weights are still predicted independently. Lin and Murray [2005] evaluate different combinations of expansion terms but assume a fixed weight of each term in the combination. In contrast, our learning procedure directly estimates full model parameterization over all terms.

As in Lease et al. [2009], we adopt a model of indirect parameterization: we define some arbitrary feature space correlated with term weights and then generate term weights via a linear model over those features. In this study, we adopt the features proposed by Cao et al. [2008] and learn feature weights via an adaptation of the AdaRank algorithm [Xu and Li, 2007]. While AdaRank as proposed uses direct parameterization, we introduce a level of indirection in training: candidate parameterizations of the feature space are not used to rank documents but to generate term weights. Given those term weights, documents can then be ranked, and evaluation of this ranking leads to a new update of parameters. AdaRank is attractive in that it facilitates direct optimization of an arbitrary retrieval metric, thus avoiding metric divergence issues associated with minimizing discordant pairs [Joachims, 2002] or other surrogate metrics. While Lease et al. [2009]'s use of efficient regression enables fast iteration in feature design and scalability to a growing feature space, such regression minimizes squared error as a surrogate for retrieval accuracy. We avoid metric divergence entirely while maintaining tractable computation.

Per our participation in the Relevance Feedback track at the 2009 Text REtrieval Conference (TREC)[1], we evaluated our approach on the newly crawled ClueWeb09 Dataset[2]. In particular, we use the "Category B" data which includes over 425 million unique URLs and 30GB of uncompressed text. We encountered two primary challenges in applying our approach to this collection: 1) performing supervised learning without existing relevance judgements, and 2) achieving tractable feature extraction (since some features drew on novel collection-wide statistics). For model training, parameters were estimated on the smaller wt10g web collection and then ported to the larger ClueWeb09; a similar approach was applied in early TREC Terabyte Tracks [Metzler et al., 2006] before relevance judgements were available for the GOV2 collection[3]. To accelerate feature extraction, we investigated strategies for subsampling the collection while preserving collection properties in the sample.

2 Model, Features, and Sampling

This section describes the model and features used in our work. As in Lease et al. [2009], there are in fact two distinct models and sets of features we apply which must be distinguished. To rank documents, we adopt standard unigram language modeling [Ponte and Croft, 1998], in which the feature space consists of the term vocabulary and the model is parameterized by the term weights. To generate these term weights, however, another model is used: a linear model defined over an arbitrary feature space. It is this model and features to which we will focus subsequent discussion. Since a goal of our study is to compare our new estimation method to that used in previous work [Cao et al., 2008], we adopt the same feature set used in that earlier study with only minimal differences as noted.

Cao et al. define five feature templates that are each instantiated on 1) the feedback documents and 2) the entire collection. For example, the first feature template yields ϕ_0 over the feedback set of documents F and ϕ_1 over the entire collection C. For the first three feature templates, "term distributions", "co-occurrence with single query term", and "co-occurrence with pairs query terms" we used them exactly as originally defined [Cao et al., 2008]. We revise the final two feature templates as follows. We denote an expansion term by e and the query by $Q = q_0 q_1 \ldots q_k$. Each feature template is defined in terms of F; the corresponding odd-numbered feature is similarly defined over C.

Weighted Term Proximity ($\phi_6(e)$) While Cao et al. used minimum distance as the distance function, we instead use average weighted distance from the expansion term to any of the co-occurring query terms.

$$\log\left(\frac{\sum_{q_i \in Q} \sum_{D \in F} C_W(q_i, e|D) * dist(q_i, e|D)}{\sum_{q_i \in Q} \sum_{D \in F} C_w(q_i, e|D)}\right)$$

where $dist(q_i, e|D)$ is the average number of terms between q_i and e in D.

Document Frequency for query terms and the expansion term together ($\phi_8(e)$) This feature provides information about the frequency of the expansion term occurring with the entire query in the set of the documents. As in [Cao et al., 2008], 0.5 is used as a smoothing factor. The corresponding feature in [Cao et al., 2008] appears to use the floor function after adding the smoothing factor, which seems to obviate the purpose of smoothing. Our interpretation is to instead apply the floor after the log

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

[2] http://boston.lti.cs.cmu.edu/Data/clueweb09

[3] http://ir.dcs.gla.ac.uk/test_collections

operation, effectively binning the feature values.

$$\lfloor \log \left(|\{D \in F \mid \forall q_i \in Q : q_i \in D \land e \in D\}| + 0.5 \right) \rfloor$$

One challenge in applying Cao et al.'s method to very large document collections like ClueWeb09 is that feature extraction for collection-wide statistics can be quite slow. To accelerate this process, we employ uniform sampling to approximate collection statistics. Sample size was selected to be roughly the size of the wt10g collection to support efficienct collection while maintaining robust statistics.

3 Estimation

Our work on supervised estimation of feedback term weights was inspired by Cao et al.'s work [2008], and we begin this section by reviewing their approach. Following this we describe our approach, and how we adapted the AdaRank algorithm [Xu and Li, 2007] for this purpose. We also discuss technical challenges encountered and our strategies for addressing them.

Cao et al. distinguish three types of expansion terms: good, neutral, and bad. These categories are defined by the impact each has on retrieval accuracy when its terms are used to expand the query. Expanding by good terms improves retrieval accuracy, expanding by neutral terms has no effect, and expanding by bad terms hurts accuracy. To label each term's category, it is added to the original query with a small fixed weight, and then the query is run and evaluated. By following this process, training data is created for learning a supervised binary classifier (neutral and bad terms are combined into one group). Expansion is then performed by independently predicting whether or not each term is good, then weighting that term by the classifier's confidence of its goodness. When integrated into the new query via either relevance modeling [Lavrenko and Croft, 2001] or the mixture model feedback [Zhai and Lafferty, 2001], superior accuracy is achieved in comparison to what either achieves using unsupervised estimation. Note that: 1) the goodness of each term in their study is determined independently, 2) this is based on a single trial with a fixed weight, and 3) their learning procedure maximizes classification accuracy rather than retrieval accuracy.

Inspired by their success, we investigate an alternative learning procedure by which retrieval accuracy is directly maximized and the interaction between feedback terms is directly modeled. As in Lease et al. [2009], term weights are generated by a linear model defined over an arbitrary feature space. Given a candidate parameterization of feature weights, our meta-training algorithm is as follows:

1. generate term weights
2. perform retrieval
3. measure accuracy
4. update feature weights based on retrieval accuracy

A variety of learning algorithms could be used with this scheme, such as simulated annealing, genetic algorithm, etc. Since each iteration involves running the query, which can be computaionally expensive as query and collection sizes increase, we desire an efficient learning algorithm minimizing the number of such iterations required. Learning to rank algorithms designed for ranking problems are particularly suitable. However, pairwise preference-based learning to rank algorithms are less desirable because we need term-based models rather than document-based models. For example, while [Kumaran and Carvalho, 2009] addressed supervised term selection using a pairwise preference-based learning to rank algorithm, they used much smaller number of terms to select compared to our case.

We chose AdaRank [Xu and Li, 2007] for the following reasons. It directly optimizes retrieval performance metrics, thus avoiding metric divergence. This allows document ranking to be performed via a traditional term-based retrieval model. AdaRank's simplicity also lends itself easily to customization for our particular training setting. While AdaRank was designed for parameter estimation in learning to rank models, our learning scenario instead introduces a new wrinkle via adding a layer of indirection:

```
Input: $S = \{(q_i, \mathbf{e_i})\}_{\mathbf{i=1}}^{\mathbf{m}}$ where $\mathbf{e_i}$ is a set of expanded terms for query $q_i$
Initialize $P_1(i) = 1/m$
For $t = 1 : T$
  - Create weak expander $h_t = h_{\hat{j}}$ by $\hat{j} = \max_j \sum_{i=1}^{m} P_t(i) E(R(h_j(q_i, \mathbf{e_i})))$
  - Choose $\alpha_t$ by $\alpha_t = \frac{1}{2} \ln \frac{\sum_{i=1}^{m} P_t(i)[1+E(R(h_t(q_i,\mathbf{e_i})))]}{\sum_{i=1}^{m} P_t(i)[1-E(R(h_t(q_i,\mathbf{e_i})))]}$
  - Create expander $f_t$ by $f_t = \sum_{k=1}^{t} \alpha_k h_k$
  - Update $P_{t+1}$ by $P_{t+1}(i) = \frac{\exp[-E(R(f_t(q_i,\mathbf{e_i})))]}{\sum_{j=1}^{m} \exp[-E(R(f_t(q_j,\mathbf{e_j})))]}$
End For
Output expander model: $f_T$
```

Figure 1: AdaRankT Algorithm. E is a retrieval evaluation measure function and R is a retrieval algorithm, e.g., the query-likelihood or Markov Random Field (MRF) model.

parameters are not used to rank documents but to generate term weights which are themselves used to rank documents. Consequently, we revise the training algorithm as illustrated in Figure 1 and refer to this procedure as AdaRankT (i.e., for term-based modeling).

AdaRankT can be distinguished from AdaRank as follows:

1. Our weak expander h_k generates expanded queries using term-based feature ϕ_k. That is, we construct a structured query with expanded term candidates using the feature values as their term weights as shown in Figure 3. However, since using all term candidates for a query sounds infeasible, we select only top M terms of all candidates by the feature values. In this work, we set $M = 100$. This expander is evaulated by retrieval results of running a conventional retrieval algorithm (e.g., query likelihood model or markov random field model) with the expanded query.

2. For weak expander selection, we see which weak expander performs best under wieght distribution P_{t-1} by running all expanded queries based on each term-based features. Since this expensive process is repeated every round, it can save time to keep search results by each weak ranker before running the AdaRankT algorithm.

3. We do not update a ranking model that is a linear model of weak rankers but a term weight model which is a linear model of term-based features. We compute a weight α_t based on a result by a weak expander h_t in the same manner as AdaRank. However, α_t plays a role of weights for term-based feature k used.

4. According to the above modifications, training data weight P is updated based on ranking results by queries expanded by linear term weight model f at round t.

One point of note is that AdaRank assumes that features positively correlate with retrieval accuracy and can only learn positive weights. However, one of our important features is actually negatively correlated, modeling discrepancy between feedback-set features and collection features. Consequently, we use $(1 - \phi)$ instead of ϕ for the collection features.

A practical challenge we encountered with AdaRank is as follows. Imagine AdaRank picks up a weak ranker based on feature ϕ_i. AdaRank decreases the weight of queries for which the selected weak ranker shows good performance. In other words, training weight distribution P is updated based on the search results by the weak ranker. Ideally, at the next round, AdaRank can be expected to select one of weak rankers based on a feature other than ϕ_i. However, ϕ_i can be dominant enough to be selected again. Then, the ranking model still depends on only a single weak ranker and the same results are returned. Therefore, P is not updated and AdaRank cannot choose other weak rankers forever.

Since AdaRankT follows AdaRank except for some modifications, we cannot avoid this problem. Instead, we tweak the algorithm as follows:

1. If the same feature is selected for the first two rounds, enqueue the strong feature and learn with only weak features.
2. Repeat (1) until any dominant feature does not appear
3. Dequeue and add back a removed feature to the remaining weak feature set. Then, learn with the set and preset the learned model into an initial model for the next iteration.
4. Repeat (3) until all features are added back

In our experiments, feature ϕ_1 was the dominant feature and always selected for the first two rounds. After the tweak, as a final model, we recovered meaningful feature weights for four features. ϕ_1 remained the most dominant while ϕ_2, ϕ_3, and ϕ_9 provided some significant contribution. This learned model achieved better performance compared to using only ϕ_1. Although previous work did not indicate the relative importance of the various features used [Cao et al., 2008], we were surprised to find only a few of the features extracted were actually assigned any weight during training.

4 The Relevance Feedback Track

Our study was conducted in the context of the 2009 TREC Relevance Feedback (RF) Track, which explores the interaction of different feedback document selection strategies with different algorithms for incorporating such feedback. The RF Track involved two distinct phases. In Phase 1, participating teams identified five documents per query to be assessed by NIST for relevance. If teams had multiple selection strategies, up to two sets of five documents per query could be submitted. In Phase 2, teams evaluated retrieval accuracy under their respective systems using different feedback document sets.

For Phase 1, we ranked documents according to two baseline systems:

- A run based on the Query Likelihood (QL) unigram model [Ponte and Croft, 1998]
- A run based on the Markov Random Field Model (MRF) [Metzler and Croft, 2005]. Weights for term, ordered, and unordered components were set to 0.80, .015, and 0.05, respectively. Due to time constraints, we did not have time to tune the weights.

To select documents for assessment, we used the following algorithm to select documents which 1) exhibited large disagreement between the two runs and 2) are less certain of being relevant and thus more likely to provide useful information via assessment.

Suppose we have ranked lists A and B.

1. Choose up to 5 best ranked documents in list B such that each document d fulfills the following criteria:
   - d does not appear in A, or
   - d's rank in A is worse than in B
2. If there are less than 5 documents found using the previous step, fill the remaining space with any document d from B that fulfills the following criteria:
   - d's rank is worse than rank 5, and
   - d has equal rank in both A and B
3. Finally, if there are still less than 5 documents found using the previous steps, fill the remaining space with any document d such that:
   - d's rank is worse than rank 5, and
   - d is not already in the list

Using this algorithm, we submitted two sets of feedback documents for Phase 1. The **UMas.1** feedback set was produced with list A coming from the QL run and B from the MRF run. The **UMas.2** feedback set used the MRF run for list A and the QL run for list B.

For phase 2, we employed our supervised learning method given different input sets of feedback documents produced by RF track participants in Phase 1. We used only those documents assessed as relevant; we leave weighting of feedback terms from non-relevant documents for future work.

As mentioned earlier, since the ClueWeb09 collection used in the RF track does not have existing relevance judgements, we train our model on the wt10g collection instead and port the learned parameters to perform retrieval on ClueWeb09. As such, one source of potential error in our experiments is mismatch between train and test collections. The wt10g collection contains 1.7 million pages and 140,470 relevance judgments for 150 topics.

5 Evaluation

This section presents results and analysis of our retrieval experiments. Retrieval accuracy on the ClueWeb09 (Category B) collection was evaluated for eight runs: a baseline run (the MRF run used in Phase 1 with additional pseudo-relevance feedback performed Lavrenko and Croft [2001]) and seven feedback runs based on different feedback sets from the track participants. For each feedback set, we performed query expansion using our learned AdaRankT model and then ran a mixture of MRF and expanded queries to be comparable to the base run. Table 1 shows the 7 assigned feedback sets and the number of topics which include positive feedback, i.e. feedback sets containing at least one relevant document. For topics with no relevant document, the baseline ranking was used.

We performed retrieval experiments using Indri [Strohman et al., 2005]. Figure 3 shows an example expanded query expressed in the Indri query language. Table 3 shows retrieval accuracy achieved with each of the seven sets of feedback documents. To provide some intuition as to what the expanded queries look like, Table 2 shows examples of the top ten weighted expansion terms selected by our method when using the `ilps.1` feedback set. Expansion sets generally appear to be semantically cohesive.

A question that arose during analysis is whether having a feedback set with more relevant documents (or documents considered to be "more relevant" would correlate with an improvement in performance. Figure 2 shows the results of our analysis, where the topics are ordered by increasing correlation coefficient. The first two sets of correlation coefficients (marked by "gr") use the graded relevance scores, therefore a feedback set can score as high as 10 (5 documents at relevance level 2). The second two sets of coefficients use flat relevance scores, meaning feedback set scores could only go as high as 5. All combinations of evaluation metric and relevance scoring behave similarly, having a fairly even spread across the spectrum of possible values. Nevertheless, when using the graded relevance which emphasizes the quality of feedback documents, more topics have positive correlation coefficients. Results suggest that at low sample sizes (i.e. 5 or less), our feedback method relies heavily on the quality of the feedback documents and less on their quantity.

5.1 Efficiency Analysis

Distributing the Indri index over a cluster of 10 CPUs, our longest run took approximately 3 hours. Table 5 shows the average run time when generating each feature from the ClueWeb_English_1 collection. Note that while the feedback set is several orders of magnitude smaller than the full collection, some of the collection-scale features were on average generated faster than their feedback set counterparts. This is due to the availability of collection-level statistics for each term provided by Indri [Strohman et al., 2005], whereas some of the feedback set statistics (e.g. number of occurrences of a term in a particular set of documents) had to be calculated on the fly. The generation times for ϕ_8 and ϕ_9 where significantly larger than the other features. This was due to our implementation effectively performing a set intersection of the posting lists of the different terms. While this may have been less efficient for the feedback-scale calculation, it was the fastest way we were aware of to calculate the feature on the whole collection. Some of our implementation could be optimized or effectively approximated, however we leave this task to future work.

6 Conclusions and Future Work

Our Phase 1 runs involved only the Query Likelihood Model and MRF Model as source runs. We wanted to perform a more thorough

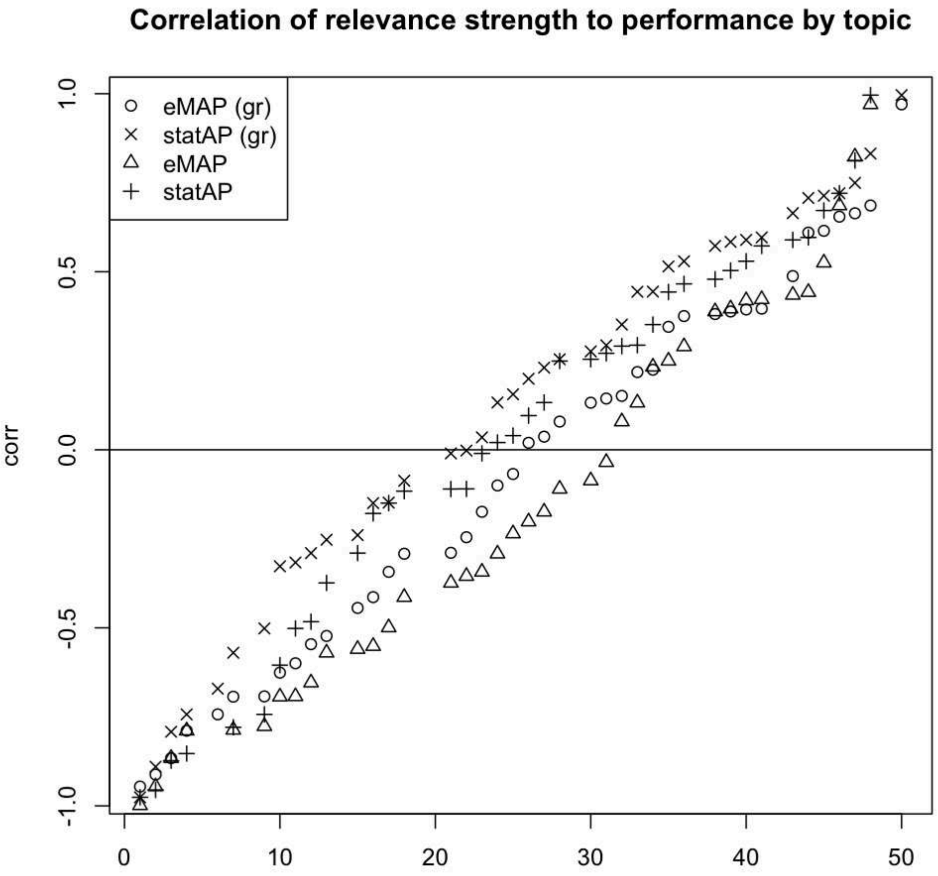

Figure 2: Correlation coefficients between number of feedback documents in a set and AdaRankT's performance based on that feedback set, ordered by increasing coefficient. When computing the coefficient for each topic, feedback sets which do not contain any relevant document were excluded.

feedback	ilps.1	PRIS.1	UCSC.2	ugTr.1	UMas.1	UMas.2	YUIR.2
#topics with pos. feedback	31	33	39	32	23	26	25

Table 1: Our assigned feedback sets and the number of topics containing at least one relevant document.

query	car parts	dinosaurs	espn sports	atari	cell phone	hoboken	dogs adoption
	auto	infraorder	disney	activision	ringtone	nj	puppy
	body	bird	abc	sega	forum	ny	pet
	lowest	extinct	channel	hardware	wireless	brook	rottweiler
	cost	jurassic	television	pioneer	palm	elizabeth	rescue
	discount	giant	network	brother	mobile	jersey	adopt
	afford	paleontologist	roster	talent	cellular	stephen	nj
	cheap	distinguish	playoff	alpha	pc	male	arrive
	low	era	basketball	ea	pocket	jr	ready
	cheapest	beast	cbs	ac	sony	lee	shelter
	putt	classification	nbc	empire	motorola	session	desperate

Table 2: Top 10 expanded terms generated by our supervised term weighting model. The query terms in expanded queries are excluded.

Metric	PRIS.1	UCSC.2	UMas.1	UMas.2	YUIR.2	ilps.1	ugTr.1	Min	Max	Avg
eMAP	.0490	.0477	.0493	.0478	.0500	.0500	.0468	.0468	.0500	.0486
StatAP	.2249	.2279	.2294	.2197	.2236	.2311	.2175	.2175	.2311	.2249

Table 3: Expected MAP (eMAP) [Carterette et al., 2006] and StatAP [Aslam and Pavlu, 2007] scores achieved by various participant runs using our selected set of Phase 1 feedback documents. A two-tailed t-test was used to test for statistically significant differences between all pairs of runs for both metrics. Only two differences were significant, both for the eMAP metric only: YUIR.2 and ilps.1 improvement in comparison to ugTr.1.

Feedback Set	eMAP		StatAP		Score
	Better	Worse	Better	Worse	
UMas.1	15	16	18	13	0.53
UMas.2	16	16	16	16	0.50

Table 4: Each set of feedback documents was used in multiple particpant runs. In comparison to other feedback sets used by those participants, how often did the given feedback set yield better or worse performance? The score is defined over both metrics as the number of better runs divided the total number of runs. With UMas.2 feedback documents, exactly half the runs did better for each metric. Runs using UMas.1 feedback documents performed slightly better than average, with better StatAP accuracy compensating for slightly worse eMAP accuracy.

```
#weight( 0.5 #weight(0.8 #combine(air travel information)
                     0.15 #combine(#1(air travel) #1(travel information) )
                     0.05 #combine(#uw8(air travel) #uw8(travel information) ) )
         0.5 #weight( 0.697082 accommodate 0.684322 caribbean 0.690662 cruise
                      0.686319 destinate 0.690842 fly 0.700626 lease
                      0.689636 premier 0.690763 resort 0.689994 route
                      0.702285 safe 0.696152 schedule 0.690543 tourism
                      0.686859 transportation 0.687628 vacation 0.691785 wine
                      ..... ) )
```

Figure 3: Example of an expanded query using Indri query language

Feature	**Time (sec)**
ϕ_0	0.014
ϕ_1	≈ 0
ϕ_2	2.555
ϕ_3	1.841
ϕ_4	0.927
ϕ_5	0.973
ϕ_6	1.977
ϕ_7	2.086
ϕ_8	28.406
ϕ_9	28.224

Table 5: Average run times in feature extraction

comparison of the results returned by performing pseudo-relevance feedback using Relevance Models Lavrenko and Croft [2001], and possibly using the MRF Model, followed by expansion using Relevance Models. The motivation behind using multiple models when searching for feedback documents would be to help characterize the query - a search engine may want to treat a query differently if two models that emphasize different aspects of a query (such as query-likelihood and MRF) return different result lists, versus another query that has a smaller perturbation between the two methods.

We considered only positive feedback instances for now. However, this may not achieve full potential from relevance feedback because documents which are classified as irrelevant can provide more clear guidelines to distinguish bad terms from good terms or neutral terms. Therefore, in future we also plan to investigate effective ways of integrating negative feedback instances in our framework.

Although our experiments seemed to favor a subset of the entire set of features available, we agree with previous work that expansion terms require a richer representation for proper selection. We would like to explore other features that may help discriminate the useful set of feedback terms from the neutral or hurtful terms contained the the feedback set.

Acknowledgments

This work was supported in part by NSF PIRE Grant No OISE-0530118, NSF IIS-0910884, NSF grant #IIS-0534383 and the Center for Intelligent Information Retrieval. Any opinions, findings and conclusions or recommendations expressed in this material are the authors and do not necessarily reflect those of the sponsors.

References

J. Aslam, E. Kanoulas, V. Pavlu, S. Savev, and E. Yilmaz. Document selection methodologies for efficient and effective learning-to-rank. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 468–475, 2009.

J. A. Aslam and V. Pavlu. A practical sampling strategy for efficient retrieval evaluation. Technical report, Northeastern University, 2007.

M. Bendersky and W. Croft. Discovering key concepts in verbose queries. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 491–498, 2008.

G. Cao, J.-Y. Nie, J. Gao, and S. Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 243–250, 2008.

B. Carterette, J. Allan, and R. Sitaraman. Minimal test collections for retrieval evaluation. In *Proceedings of the 29th annual international ACM SIGIR conference* *on Research and development in information retrieval*, pages 268–275, 2006.

N. Fuhr and C. Buckley. A probabilistic learning approach for document indexing. *ACM Transactions on Information systems*, 9(3):223–245, 1991.

T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, 2002.

G. Kumaran and V. Carvalho. Reducing long queries using query quality predictors. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 564–571, 2009.

V. Lavrenko and W. B. Croft. Relevance based language models. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 120–127, New York, NY, USA, 2001. ACM. ISBN 1-58113-331-6. doi: http://doi.acm.org/10.1145/383952.383972.

M. Lease, J. Allan, and W. B. Croft. Regression rank: Learning to meet the opportunity of descriptive queries. In *ECIR '09: Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, pages 90–101, 2009.

J. Lin and G. Murray. Assessing the term independence assumption in blind relevance feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, page 636. ACM, 2005.

Q. Mei, H. Fang, and C. Zhai. A study of Poisson query generation model for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 319–326, 2007.

D. Metzler and W. B. Croft. A markov random field model for term dependencies. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 472–479, New York, NY, USA, 2005. ACM. ISBN 1-59593-034-5. doi: http://doi.acm.org/10.1145/1076034.1076115.

D. Metzler, T. Strohman, Y. Zhou, and W. Croft. Indri at TREC 2005: Terabyte Track. In *Proceedings of the Fourteenth Text REtrieval Conference (TREC 2005)*, 2006.

J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281, 1998.

G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. Technical Report 97-881, Cornell University, 1987.

K. Sparck Jones, S. Walker, and S. Robertson. A probabilistic model of information retrieval: development and comparative experiments (parts i and ii). *Information Processing and Management*, 36:779–840, 2000.

T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: a language-model based search engine for complex queries. Technical report, in Proceedings of the International Conference on Intelligent Analysis, 2005.

J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, page 398, 2007.

C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the 10th conference on Information and knowledge management (CIKM)*, pages 403–410, 2001.