Relevance Models for Topic Detection and Tracking

Victor Lavrenko, James Allan,
Edward DeGuzman, Daniel LaFlamme, Veera Pollard, and Stephen Thomas

Center for Intelligent Information Retrieval
Department of Comupter Science
University of Massachusetts, Amherst, MA 01003

ABSTRACT

We extend relevance modeling to the link detection task of Topic Detection and Tracking (TDT) and show that it substantially improves performance. Relevance modeling, a statistical language modeling technique related to query expansion, is used to enhance the topic model estimate associated with a news story, boosting the probability of words that are associated with the story even when they do not appear in the story. To apply relevance modeling to TDT, it had to be extended to work with stories rather than short queries, and the similarity comparison had to be changed to a modified form of Kullback-Leibler. We demonstrate that relevance models result in very substantial improvements over the language modeling baseline. We also show how the use of relevance modeling makes it possible to choose a single parameter for within- and cross-mode comparisons of stories.

1. INTRODUCTION

Topic Detection and Tracking (TDT) is a research program investigating methods for automatically organizing news stories by the events that they discuss. TDT includes several evaluation tasks, each of which explores one aspect of that organization–i.e., splitting a continuous stream of news into stories that are about a single topic ("segmentation"), gathering stories into groups that each discuss a single topic ("detection"), identifying the onset of a new topic in the news ("first story detection"), and exploiting user feedback to monitor a stream of news for additional stories on a specified topic ("tracking").

Another TDT evaluation task, Link Detection, requires determining whether or not two randomly selected stories discuss the same topic. Unlike the other tasks that have value in and of themselves, Link Detection is a a component technology: it can be used to address each of the other tasks.[1] For example, in order to recognize the start of a new topic, a candidate story might be compared to all prior stories to see whether the topic appeared earlier. Similarly, tracking stories on a specified topic can be done by comparing each arriving story to the user-supplied list of on-topic stories.

The core of most approaches to Link Detection (and therefore to most of the TDT tasks) is comparing word overlap between the two stories: the more words that are in common, the more likely it is that the two stories are on the same topic. This method is the basis of everything from vector space approaches [4, 17] to statistical language model [19, 12] techniques. As in the field of Information Retrieval, much research focuses on techniques for selecting which words to compare, how they should be weighted, and how best to compare the sets of weighted words.

Another research idea that has become common in TDT is expanding the words in a story to include other, strongly related words. The idea is to increase the likelihood that two stories on the same topic will have important overlapping words. Vector space models use query expansion techniques such as Local Context Analysis [18] to expand the list of words [16]. More theoretically grounded approaches such as statistical language modeling implicitly include related words from a background model as part of the smoothing process [19, 12].

In this paper we explore this technique of expanding the set of words associated with a story using "relevance models," a theoretical extension of statistical language modeling that was developed for the task of document retrieval. In the next section we outline the basic ideas of relevance models. In Section 3 we present how relevance models were adapted to the TDT tasks, where stories are compared to each other rather than a query to a document, and where the comparison method is therefore different. Section 4 describes the experiments that were done and shows how relevance models improve effectiveness on the link detection task. In Section 4.5 we briefly explore how this expansion process is quite different depending on whether the stories come from within a single source and modality, or across them. Section 5 draws conclusions and speculates on future work in this area.

2. RELEVANCE MODELS

Lavrenko and Croft [11] define "relevance model" to be a mechanism that determines the probability $P(w|R)$ of observing a word w in a document that is relevant to a query,

[1] In fact, such relationships exist between several of the tasks. The tracking task can be a basis for the other tasks also [3], though unlike link detection, it has clear value on its own.

.

where R represents the class of documents that are relevant to the query. The difficult aspect of relevance models is estimating the model in the absence of significant amounts of data—typically the system has available only the query and a set of documents without relevance judgments.

One way to address that problem is to make the assumption that in the absence of any training data and given a query $Q = q_1 q_2 \ldots q_k$, we can approximate $P(w|R)$ by $P(w|q_1 q_2 \ldots q_k)$. That is, by the probability of co-occurrence between the query and the word.

To implement the ideas behind relevance models, a system follows the following steps:

- Use the query, Q, to retrieve a set of highly ranked documents, R_Q. This step yields a set of documents that contain most or all of the query words. The documents are, in fact, weighted by the probability that they are relevant to the query—i.e., $P(D|Q), D \in R_Q$.
- For each word in those documents, calculate the probability that the word occurs in that set. That is, calculate $P(w|R_Q)$, a value that is used to approximate $P(w|R)$:

$$P(w|R_Q) = \sum_{D \in R_Q} P(w|D)P(D|Q) \quad (1)$$

As with most language modeling approaches, $P(w|D)$ is calculated using a maximum likelihood estimate smoothed with the background model:

$$\begin{aligned} P(w|D) &= \lambda P_{ml}(w|D) + (1-\lambda)P_{bg}(w) \\ &= \lambda \frac{\mathrm{tf}_{w,D}}{|D|} + (1-\lambda)\frac{\mathrm{cf}_w}{\text{coll.size}} \end{aligned} \quad (2)$$

Here, $\mathrm{tf}_{w,D}$ is the number of times the word w occurs in the document D, cf_w is the total number of times w occurs in a large background collection, and coll.size is the total number of words in that background collection. The value of λ as well as the number of documents to include in R_Q are determined empirically from training data. For all experiments reported here, R_Q is 30. Values of λ are reported below.

3. RELEVANCE MODELS IN TDT

The relevance model described above was built from a small query and then used to estimate the probability that documents are relevant to the query. In TDT's link detection task two stories are being compared to decide if they are on the same topic. We will do this by building a relevance model for *each* story, where that model is intended to capture the topic of the stories. Because we are starting from an entire story rather than from a short query, the estimation must be done slightly differently.

Once the two models are created, we decide whether they discuss the same topic by comparing the models directly. We found it helpful to use a modified form of the Kullback-Leiblar divergence

Finally, we needed to explore the impact of different modalities on the comparisons. In most retrieval tasks, the collection being searched contains a mostly homogeneous set of documents. TDT collections, on the other hand, come from newswire text, speech recognition output, closed captioning, machine translation, and combinations of all of those. Comparisons within and across modalities can have significant impact on the results. We discuss below how we addressed that problem.

3.1 Building the topic models

In order to build a topic (relevance) model for a story, we start with the process described above. The story, $S = q_1 q_2 \ldots q_k$, is used as a "query" into all training stories in the news. Each training story, D, is therefore ranked by $P(D|S)$ or $P(D|q_1 \ldots q_k)$. Unfortunately, if those probabilities are used directly, they are generally forced to zero because there are so many terms in the story (i.e., k is large). To see why that is the case, consider how we calculate the probability:

$$P(D|q_1 \ldots q_k) = \frac{P(q_1 \ldots q_k|D)P(D)}{P(q_1 \ldots q_k)}$$

The denominator $P(q_1 \ldots q_k)$ is a constant across different documents D, and following [11] we picked a a uniform prior $P(D)$. Therefore only the probability of the query given the document $P(q_1 \ldots q_k|D)$ has an impact on the value of the posterior. We usually assume independence of the query words and calculate:

$$P(D|q_1 \ldots q_k) \sim P(q_1 \ldots q_k|D) = \prod_{i=1}^{k} P(q_i|D)$$

When k is larger than a few terms, the resulting product becomes very small (recall that $P(q_i|D) < 1$). As a result, we often get floating point underflow and—more importantly—the probability only has a reasonable value for the highest-ranked story. That is, $P(D|q_1 \ldots q_k)$ is driven to zero for all but the highest ranked story, which is almost always the original document $S = q_1 \ldots q_k$. If that happens, then the value of relevance modeling disappears: no stories are mixed into the model.

To address this problem, we "flatten" the probability as $P(q_1 \ldots q_k|D)^{\frac{1}{k}}$. Taking the k^{th} root avoids both problems listed above. This is a heuristic adjustment which we hope to circumvent in our future work on relevance-based models.

3.2 Measuring Topic Similarity

Once we have built models for each topic, we need to compare the models to determine the chance that they represent the same topic. Given two stories, S_1 and S_2, assume that their relevance models are M_1 and M_2, respectively. If we were to parallel the information retrieval use of relevance models, we might calculate either $P(S_1|M_2)$ or $P(S_2|M_1)$ or possibly the average of the two.

However, given that we have two models that were estimated from similar amounts of data (S_1 or S_2) we can instead compare the models directly. The Kullback-Leibler divergence is a standard way to compare two probability distributions, defined as:

$$D(M_1||M_2) = \sum_w P(w|M_1) \log \frac{P(w|M_1)}{P(w|M_2)}$$

KL divergence is asymmetric, which is unacceptable as a link detection metric. We compute a symmetric version by summing the divergence in both directions: $D(M_1||M_2) + D(M_2||M_1)$. Since Kullback-Leiblar divergence is a measure

of dissimilarity of the two distributions, we use the negation of the above quantity to measure similarity.

This yields a reasonable approach, but has the problem that if the models are very ambiguous—e.g., if relevance modeling failed and created a model that looks too much like general English—their matching has very little significance. That is, if the two models are both of general English, it is not valuable that they are identical since they do not describe a topic. To address this problem, we leverage a notion of query clarity [8], the KL divergence between a distribution and general English. A distribution is clear (or focused) if it is very unlike general English and unclear if it is identical to general English.

The non-symmetric version of our topic similarity measure is therefore: $[-D(M_1||M_2) + \text{Clarity}(M_1)]$ That is, the degree to which M_2 and M_1 are similar, increased to the extent that M_1 is a clear model that differs from general English. After a very simple algebraic manipulation, the similarity measure can be written as:

$$\sum_w P(w|M_1) \log \frac{P(w|M_2)}{P(w|GE)} \quad (3)$$

In this form it bears strong resemblance to the length-normalized log-likelihood ratio, which has been used by a number of TDT participants [12, 4]. Note that adding clarity has resulted in the denominator that plays a role similar to the role of *idf* in document retrieval. To get the final similarity measure, we calculate the same quantity the other way (swapping M_1 and M_2) and add them together.

4. APPLYING RELEVANCE MODELS

In this section we evaluate performance of relevance models, as described above, on the Link Detection task of TDT. First, we describe the experimental setup and the evaluation methodology. Then we provide empirical support for the choice of parameters in our system. Finally we show that relevance models significantly outperform simple language models, as well as other heuristic techniques.

4.1 Experimental Setup

4.1.1 Dataset

All of the following experiments were performed on a 4-month subset of the TDT2 [7] dataset. The corpus contains 40,000 news stories totaling around 10 million words. The news stories were collected from six different sources: two newswire sources (Associated Press and New York Times), two radio sources (Voice of America and Public Radio International), and two television sources (CNN and ABC). The stories cover January through April of 1998. Radio and television sources were manually transcribed [7] at closed-caption quality. In a pre-processing stage, the stories were stemmed and 400 stop-words from the InQuery [5, 2] stoplist were removed.

4.1.2 Topics

TDT is concerned with detecting and organizing the topics in news stories. Human annotators identified a total of 96 topics in the TDT2 dataset, ranging from 1998 Asian financial crisis, to the Monica Lewinsky scandal, to an execution of Karla Faye Tucker. Each topic is centered around a specific event, which occurs in a specific place and time, with specific people involved. 56 out of these 96 topics are sufficiently represented in the first four months of 1998 and will form the basis for our evaluation.

4.1.3 Evaluation Paradigm

The system is evaluated in terms of its ability to detect the pairs of stories that discuss the same topic. A total of 6363 story pairs were drawn from the dataset (according to the official TDT2 sampling). 1469 of these were manually [7] judged to be on-target (discussing the same topic), and 4894 were judged off-target (discussing different topics). During evaluation the Link Detection System emits a YES or NO decision for each story pair. If our system emits a YES for an off-target pair, we get a False Alarm error; if the system emits a NO for on-target pair, we get a Miss error. Otherwise the system is correct. Link Detection is evaluated in terms of the decision cost [9], which is a weighted sum of probabilities of getting a Miss and False Alarm:

$$Cost = P(Miss)C_{Miss} + P(FA)C_{FA}$$

In current evaluations of Link Detection, C_{Miss} is typically set to 10×0.02, and $C_{FA} = 1 \times 0.98$. Note that by always answering YES a system would have no misses and therefore a cost of 0.2 (similarly, always answering NO guarantees a cost of 0.98). To penalize systems for doing no better than a simple strategy like that, the cost is normalized by dividing by the minimum of those two values (here, 0.2). A normalized cost value near or above 1.0 reflects of a system of marginal value. An operational Link Detection System requires a threshold selection strategy for making YES / NO decisions. However, in a research setting it has been a common practice to ignore on-line threshold selection and perform evaluations at the threshold that gives the best possible cost. All of our experiments report the minimum normalized detection cost: $Cost_{min}$.

4.2 Value of Clarity-adjusted KL

In Section 3 we describe a clarity-adjusted KL divergence, and provide an argument for why we believe this measure may perform better than straight KL divergence. To evaluate the value of clarity adjustment we perform a simple experiment without constructing relevance models. Given a pair of stories A and B we construct maximum likelihood language models of each story, smooth it with the background model, as described in equation (2), and measure divergence. We consider four different divergence measures:

1. simple KL divergence: $KL_1(A, B) = D(A||B)$
2. symmetric version: $KL_2(A, B) = D(A||B) + D(B||A)$
3. clarity-adjusted: $KL_{c,1}(A, B) = D(A||B) - Clarity(A)$
4. symmetric: $KL_{c,2}(A, B) = KL_{c,1}(A, B) + KL_{c,1}(B, A)$

Figure 1 shows the minimum detection cost ($Cost_{min}$) of the four measures as a function of the smoothing parameter λ from equation (2). We observe that clarity-adjusted KL leads to significantly lower errors for all values of λ. Clarity adjustment also leads to smaller dependency on λ, which makes tuning easier. We also note that for both simple and adjusted KL, we get significantly better performance by using symmetric divergence. The best performance $Cost_{min} = 0.1057$ is achieved by using the symmetric version of clarity-adjusted KL when $\lambda = 0.2$. This

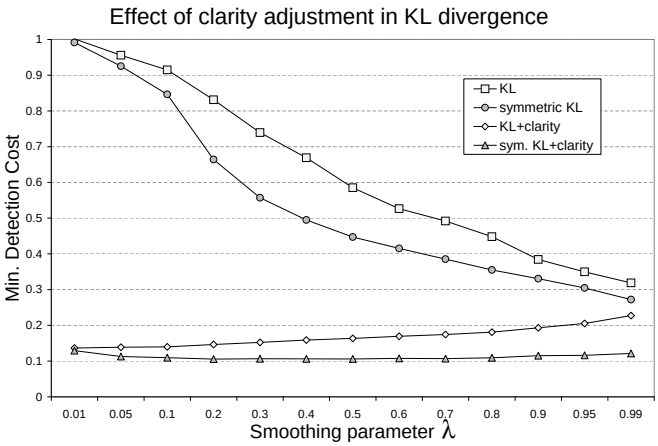

Figure 1: Clarity adjustment leads to significantly lower error rates. Symmetric versions of KL perform better than asymmetric versions. Symmetric KL with clarity is best and most stable with respect to the smoothing parameter λ.

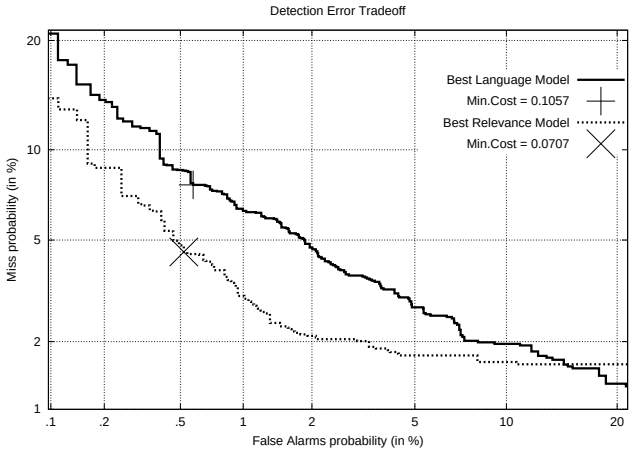

Figure 2: Relevance Models noticeably outperform the baseline for all threshold settings in the region of interest. Minimum Detection Cost is reduced by 33%

performance will be used as a baseline in later comparisons with relevance models. The baseline is competitive with the state-of-the-art results reported in the official TDT evaluations [14].

4.3 Relevance Model Performance

Now we compare our baseline to the performance we can achieve with relevance models. Given a pair of stories A and B, we construct a relevance model from each story as described in Sections 2 and 3. For efficiency purposes, each story was reduced to 30 words with highest frequencies, and these words were used as a query Q to retrieve a set of related stories R_Q. The set R_Q was limited to contain 30 highest-ranked documents. The relevance model was constructed from this set of documents according to equation (1). We use symmetric clarity-adjusted KL as a measure of divergence between the two resulting relevance models. The smoothing parameter λ is set to 0.999. Empirical justifications for these parameter settings are described in section 4.4.

Figure 2 shows the *Detection Error Tradeoff* (DET) [13] curve for the performance of relevance models, compared to the best language modeling baseline. A DET curve is a plot of Miss and False Alarm probabilities as a function of a sliding threshold. The point on each curve marks the optimal threshold, and the corresponding minimum cost $Cost_{min}$. Note that the values are plotted on a Gaussian scale and that the axes only go up to 20% Miss and False Alarm; the full-range DET curves are presented in Figure 6. We observe that a relevance modeling system noticeably outperforms the baseline for almost all threshold settings. The improvements are particularly dramatic around the optimal threshold. The minimum cost is reduced by 33%. Outside of the displayed region, on the high-precision end ($FalseAlarm < 0.01\%$), the relevance modeling system noticeably outperforms the baseline. On the very high-recall end ($Miss < 1.5\%$), the baseline performs somewhat better.

The results in Figure 2 were achieved by careful tuning of parameters on the 4-month subset of the TDT-2 corpus and do not represent a blind evaluation. However, the same parameters were used in the official TDT 2001 blind evaluation on the 3-month TDT-3 corpus. The results in that case were comparable to those described above (see Figure 7 for details). The system based on Relevance Models significantly outperformed a state-of-the-art vector-space system (cosine with Okapi *tf.idf* weighting). The normalized minimum cost was reduced from 0.27 to 0.24. This suggests that our parameter settings generalized reasonably well to the new dataset and the new set of topics.

4.4 Parameter Selection

In this section we provide empirical justification for the values of parameters we used in the estimation of relevance models. We explore three parameters: the number of words we used to represent the document, the size of the retrieved set R_Q, and the smoothing parameter λ.

4.4.1 Query size

Efficiency issues prompted us to represent a document by 30 words with highest frequencies in that document. Ideally, we would like to be able to use all the words in the document, however running the whole document as a query is computationally very expensive, while running a 30-word query is quite feasible. To show that 30 words is a reasonable value, we look at resulting precision in the set R_Q for different sizes of R_Q. Figure 3 shows the precision as a function of the number of words in the query. Precision is the proportion of documents in R_Q that discuss the same topic as the query document. We show precision for the sets R_Q of size 5, 10, 15, 20 and 30. We observe that in all cases precision does not improve significantly if we use more than 30 words. If we use fewer than 30 words, the precision degrades noticeably. In the remaining experiments we use 30-word queries. Note that stopwords have been removed from the documents prior to selecting high-frequency words.

4.4.2 Size of retrieved set

In theory, estimating a relevance model involves computing equation (1) over all the documents D in the dataset, since $P(D|Q)$ is never zero due to the smoothing. In practice this is very expensive and unnecessary. For the vast majority

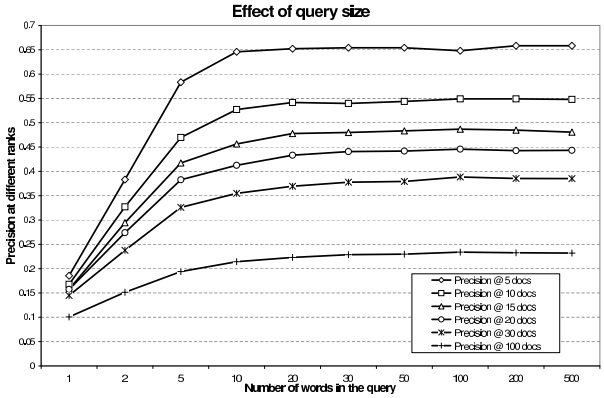

Figure 3: Using more than 30 words to represent the document does not lead to improved precision in the set R_Q. The result is consistent over various sizes of R_Q.

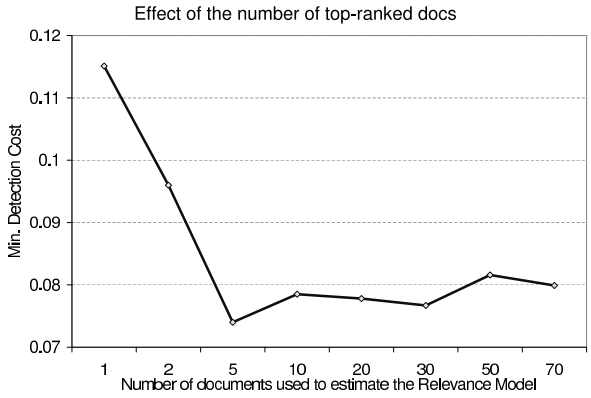

Figure 4: Using anywhere between 5 and 70 top-ranked documents as R_Q results in consistently good performance.

of documents, $P(D|Q)$ is very close to zero, and including it in the summation will have little effect. For this reason we consider limiting R_Q to the n documents with the highest $P(D|Q)$. Figure 4 shows relevance models performance as a function of n (the smoothing parameter λ was fixed at 0.9). We observe that using anywhere between 5 and 70 documents with highest $P(D|Q)$ results in good performance. Using fewer than 5 or more than 100 documents has adverse effects. We settled on using 30 top-ranked documents.

4.4.3 *Smoothing*

Smoothing is a critical component of any system based on statistical language modeling. Numerous studies [6, 10, 15] have shown that smoothing has a very strong impact on performance of information retrieval systems. In Section 4.2 we observed that the smoothing from equation (2) has a strong effect on performance of maximum likelihood models. Interestingly, smoothing had a very unusual effect with relevance models. We obtained best performance with λ very close to 1, i.e. with almost no smoothing of the document models. This result is somewhat unexpected, and deserves a brief explanation.

According to Zhai and Lafferty [10], smoothing plays a dual role in applications of statistical language models to Information Retrieval. First, smoothing ensures non-zero probabilities for every word under a document model, and acts as a variance-reduction technique. Second, smoothing has an *idf*-like effect on document scoring. Both of these roles are captured by different mechanisms in our model. When we estimate a Relevance Model for some story, we mix together neighboring document models (equation 1). This results in non-zero probabilities for many more words than actually occur in the original story, so there is less value in smoothing to avoid zeros. Also, as mentioned in the end of section 3.2, clarity adjustment on KL divergence has an effect similar to *idf*.

4.5 Cross-modal evaluation

An essential part of TDT is being able to deal with multiple sources of news. The TDT2 corpus that was used in our experiments includes news from six different sources. Two of these (Associated Press and New York Times) are printed sources, the other represent broadcast news, which are transcribed from audio signal. Spoken text has very different properties compared to written sources, and part of the TDT challenge is development of the algorithms that can cope with source-related differences in reporting. To determine how well our algorithms perform on different source conditions, we partitioned the set of 6363 pairs into three subsets:

1. 2417 pairs where both stories come from a broadcast source; this set will be labeled "BN" (broadcast news)

2. 1027 pairs where both stories come from a printed source; this set will be labeled "NW" (newswire)

3. 2919 pairs where one story is from a broadcast source and the other from the printed source; this set will be labeled "NWxBN"

Figure 5 shows performance of the baseline and the relevance modeling systems on the three subsets we described. Performance is shown as a function of the smoothing parameter λ. First we observe that performance varies very significantly from one subset to another. Interestingly, both systems perform best on the "NWxBN" condition, even though it intuitively appears to be more challenging as we are dealing with two different language styles. Another very interesting issue is the value of λ that gives the best performance. Note that for the baseline system the optimal λ value is different for every condition: "BN" is optimized near $\lambda = 0.5$, "NW" – near $\lambda = 0.05$, while "NWxBN" is optimal near $\lambda = 0.7$. This means that for the baseline system we cannot select a single value of λ which will work well for all sources. In contrast to that, for the relevance modeling system all conditions are optimized if we set λ to 0.99, or any value close to 1. This is a very encouraging result, as it shows that relevance models are not very sensitive to source conditions.

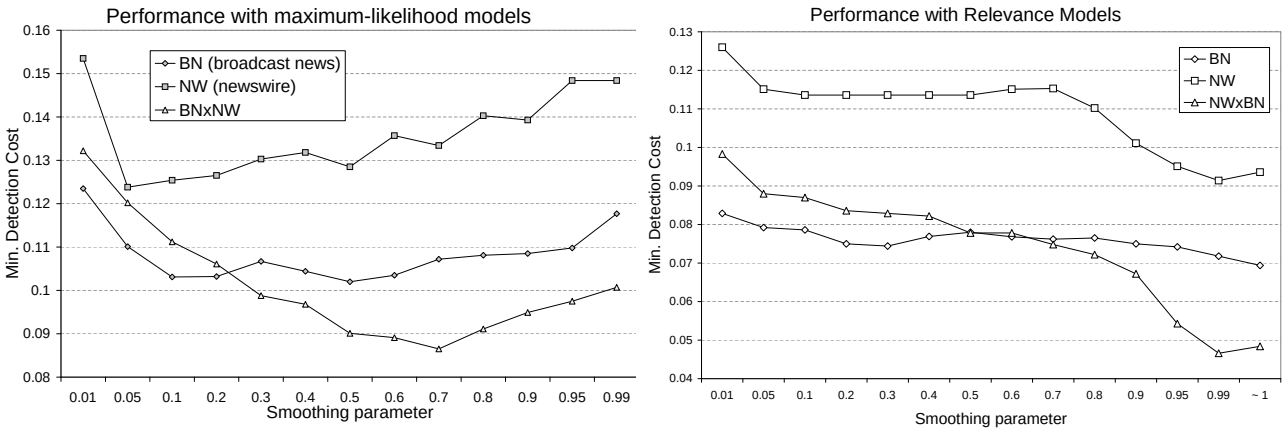

Figure 5: Performance on different source conditions. Left: baseline, optimal smoothing value is different for every condition. Right: Relevance Models, all conditions are optimized as λ approaches 1.

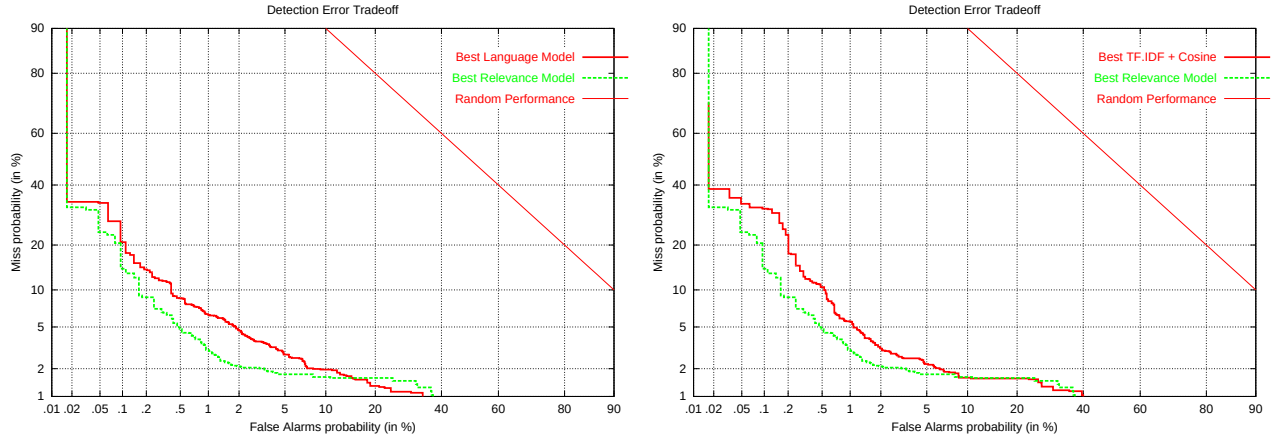

Figure 6: Performance of Relevance Models on the training data. Relevance Models with optimal parameters outperform both the optimal Language Modeling system (left), and the optimal vector-space system using Cosine with Okapi term weighting (right). Minimum Detection Cost was reduced by 33% and 25% respectively (not shown).

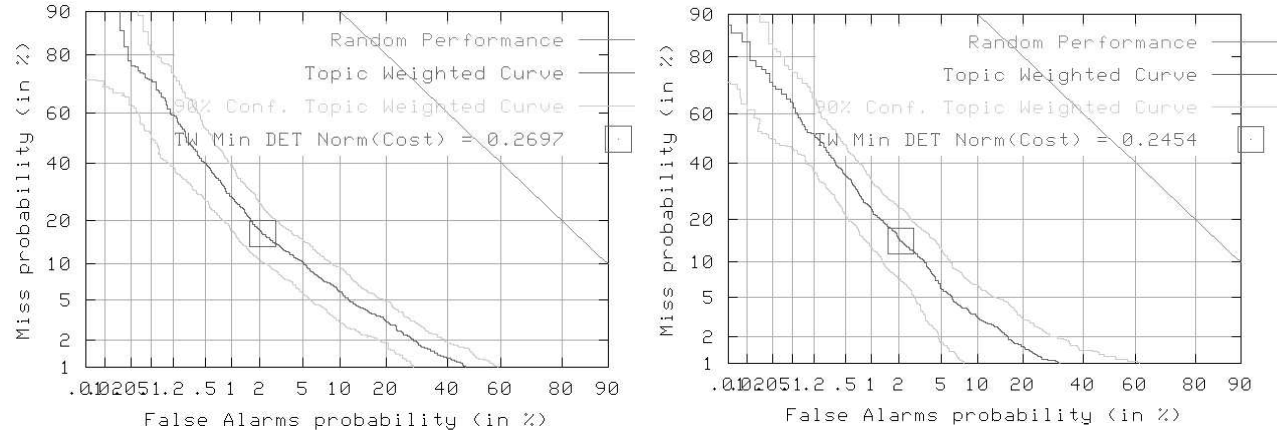

Figure 7: Performance of Relevance Models in the official TDT 2001 evaluation. All the parameters were tuned on the training dataset, and no part of the evaluation dataset was used prior to evaluation. Relevance Models (right) consistently outperform the vector-space model (left). Minimum Detection Cost is reduced by 10%.

5. CONCLUSIONS

In this work we have shown how the relevance model technique can be extended to TDT's link detection task. To do so, the models themselves had to be calculated differently to avoid the problem of very small probabilities due to large "queries". Also, we found it preferable to compare two models directly using a version of KL divergence that incorporates model clarity, or how close the model is to general English.

We demonstrated a substantial performance improvement using relevance models. The parameter selection problem was also shown to be somewhat simpler because the cost values are less sensitive to parameter changes with than without relevance models.

This effect—easier parameter selection—carried over into the problem of within- and cross-mode comparisons of stories. With relevance models, the different choices (both A, both B, or A&B) have much more similar cost values, improving the error tradeoffs.

6. FUTURE WORK

In the course of this work, we encountered a number of interesting questions that we hope to answer in our future research. For one, we were surprised by the gain we achieved by using clarity adjustment over the straight KL divergence, and would like to investigate theoretical implications of its remarkably good performance. Second, we are not satisfied with the heuristic nature of *flattening* of posterior probabilities (section 3.1), and are investigating more formal approaches.

The present work can be extended in a number of important directions. One is dictated by the multi-lingual nature of TDT: a Link Detection system should be capable of dealing with stories in multiple languages. We are actively investigating techniques for estimating multi-lingual relevance models, i.e. language models that contain a mixture of English and non-English words. We are also interested in extending the framework of relevance models to the case where the stories discuss multiple topics. In this case, multiple relevance models would be formed for each story in question.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval, in part by the National Science Foundation under grant number EIA-9820309, and in part by SPAWARSYSCEN-SD grant number N66001-99-1-8912. Any opinions, findings and conclusions or recommendations expressed in this material are the authors and do not necessarily reflect those of the sponsors.

7. REFERENCES

[1] J. Allan, editor. *Topic Detection and Tracking: Event-based Information Organization*. Kluwer Academic Publishers, Boston, 2002.

[2] J. Allan, M. Connell, W. Croft, F. Feng, D. Fisher, and X. Li. INQUERY and TREC-9. In *Proceedings of the Text Retrieval Conference (TREC-9)*, pages 551–577. NIST, 2001.

[3] J. Allan, V. Lavrenko, and H. Jin. First story detection in TDT is hard. In *Proceedings of the Conference on Information and Knowledge Management (CIKM)*, pages 389–396, 2000.

[4] J. Allan, V. Lavrenko, and R. Swan. Explorations within topic tracking and detection. In Allan [1], pages 197–224.

[5] J. Broglio, J. Callan, and W. B. Croft. INQUERY system overview. In *Proceedings of the Tipster Text Program (Phase I)*, pages 47–67, San Francisco, 1994. Morgan Kaufmann.

[6] S. F. Chen and J. T. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the ACL*, 1996.

[7] C. Cieri, S. Strassel, D. Graff, N. Martey, K. Rennert, and M. Liberman. Corpora for topic detection and tracking. In Allan [1], pages 33–66.

[8] W. B. Croft, S. Cronen-Townsend, and V. Lavrenko. Relevance feedback and personalization: A language modeling perspective. In *Proceedings of the DELOS-NSF Workshop on Personalization and Recommender Systems in Digital Libraries*, pages 49–54, 2001.

[9] J. G. Fiscus and G. R. Doddington. Topic detection and tracking evaluation overview. In Allan [1], pages 17–31.

[10] J. Lafferty and C. Zhai. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings on the 24th annual international ACM SIGIR conference*, pages 111–119, 2001.

[11] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Proceedings of ACM SIGIR Conference on Research in Information Retrieval*, pages 267–275, 2001.

[12] T. Leek, R. Schwartz, and S. Sista. Probabilistic approaches to topic detection and tracking. In Allan [1], pages 67–83.

[13] A. Martin, G. Doddington, T. Kamm, and M. Ordowski. The DET curve in assessment of detection task performance. In *EuroSpeech*, pages 1895–1898, 1997.

[14] NIST. Proceedings of the tdt 2001 workshop. Notebook publication for participants only, Nov. 2001.

[15] J. Ponte. Is information retrieval anything more than smoothing? In *Proceedings of the Workshop on Language Modeling and Information Retrieval*, pages 37–41, 2001.

[16] J. Ponte and W. B. Croft. Text segmentation by topic. In *Proceedings of the European Conference on Research and Advanced Technology for Digital Libraries (ECDL)*, pages 113–125, 1997.

[17] J. M. Schultz and M. Y. Liberman. Towards a "universal dictionary" for multi-language IR applications. In Allan [1], pages 225–241.

[18] J. Xu and W. B. Croft. Improving the effectiveness of inforamtion retrieval with local context analysis. *ACM Transactions on Information Systems (TOIS)*, 18(1):79–112, 2000.

[19] J. Yamron, L. Gillick, P. van Mulbregt, and S. Knecht. Statistical models of topical content. In Allan [1], pages 115–134.