TDT-2002 Topic Tracking at Maryland: First Experiments with the Lemur Toolkit

*Daqing He, Hyuk Ro Park, G. Craig Murray, Michael Subotin and Douglas W. Oard**

Institute for Advanced Computer Studies
University of Maryland, College Park, MD 20742

ABSTRACT

The University of Maryland submitted six topic tracking runs for the 2002 Topic Detection and Tracking evaluation. Two runs were produced using the Lemur language modeling toolkit, the remaining four were produced using an separate system coded in Perl. The Lemur runs outperformed the Perl runs on the required condition because term frequency information was better handled. Two of the Perl runs used native Arabic orthography with two-best translation based on a statistical lexicon, obtaining similar results to those obtained with the Arabic-to-English translations provided with the collection.

1. Introduction

The University of Maryland participated in the topic tracking task of the 2002 Topic Detection and Tracking (TDT) evaluation. We had two goals this year: (1) to develop a credible baseline system to support continued Arabic-English translingual detection experiments that will build on the work that we have done in the Text Retrieval Conference's (TREC) Cross-Language Information Retrieval (CLIR) track, and (2) to begin to explore the use of language models for information retrieval. We have previously participated in TDT-1998, TDT-1999 and TDT-2000, in each case building a topic tracking system around the freely available PRISE text retrieval system [8]. This year, we chose to work with the Lemur toolkit [14].

Language modeling techniques for information retrieval have received increasing attention since their introduction in 1998 [11], and they seem well suited to the topic tracking task as well [7]. The Lemur toolkit was developed jointly by the University of Massachusetts and Carnegie Mellon University to facilitate development of retrieval systems based on language models, and TDT-2002 provided us with an excellent opportunity to learn about its capabilities. We built two systems using Lemur. In the first, we indexed the collection using components from Lemur, then wrote Perl scripts to perform topic tracking. This offered a useful degree of insight into several important implementation details. We then reimplemented similar run-time processing within Lemur. We submitted results using both systems; our Lemur runs appear as "UMD2" in the official results, our Perl runs appear as "UMD3."

The topic tracking task poses three challenges that are not present in the TREC 2001/2002 Arabic-English CLIR task: (1) cross-topic score normalization, (2) cross-language score normalization, and (3) cross-source score normalization. We therefore chose to focus on score normalization in our TDT experiments. Readers are referred to our TREC-2002 CLIR track paper for our latest thinking on other aspects of Arabic/English translingual detection.

The remainder of this paper is organized as follows. Section 2 describes the tracking model we developed for our experiments, and Section 3 describes some important implementation details. We then describe the conditions that we ran, the results we obtained, and our preliminary analysis of the results in Section 4. Finally, we conclude with some thoughts on future directions that we expect this work to take in Section 5.

2. Modeling Topic Tracking

The key idea behind every approach to detection that we are aware of is to model the use of terms in previously seen on-topic and off-topic stories, and then rank newly arrived stories based on the degree to which term use in the new story matches the on-topic model. Language modeling techniques are distinguished from other approaches to detection by their use of estimation techniques originally developed in speech recognition. Most present techniques rely on unigram language models, which incorporate the same term independence assumption that underlies all bag-of-terms approaches to detection. The key questions are then: (1) what probabilities are modeled, (2) how are those probabilities estimated, and (3) how are those estimates used? In the first part of this section we develop the framework that we have implemented. We then focus specifically on score normalization and on translingual techniques in the two remaining parts of the section.

2.1. The Language Model

Several researchers have applied a language modeling framework to ad-hoc text retrieval, generally reporting promising results [3, 10, 11]. Many of these approaches rank documents according to the probability of generating a query Q by repeated sampling from document model D. Assumed term independence, document are ranked in decreasing order of:

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

An analogy could be drawn between scoring each document

*College of Information Studies and Institute for Advanced Computer Studies

based on a query in ad-hoc retrieval and evaluating each story S with respect to a topic T in topic tracking. So in a simple case, we could use Equation (1) by substituting the topic T for the query Q and the story S for the document D. Spitters and Kraaij suggested an alternate approach, however, noting that we generally have more information about a topic in the TDT topic tracking setting than we would have about a query in the ad-hoc retrieval setting because we are given 1 to 4 training stories as evidence of the user's information need [13]. We chose, therefore, to start with their approach, computing the probability of the story S being generated by the topic T:

$$P(S|T) = \prod_{w \in S} P(w|T) \tag{2}$$

A topic model built from only the on-topic training stories would be quite sparse, so we smooth the model with a background language model B using linear interpolation:

$$P(w|T) \simeq \lambda P(w|T) + (1-\lambda)P(w|B) \tag{3}$$

During system development and parameter tuning, we built the background model using the TDT-2 collection, using the TDT-3 collection for development testing. For our official submissions, we built the background model using both the TDT-2 and TDT-3 collections. We estimated these probabilities using a maximum likelihood estimate, where the model M can be a language model representing topic T or background B:

$$P(w|M) = \frac{freq(w \ in \ M)}{\sum_{w_i \in M} freq(w_i \ in \ M)} \tag{4}$$

Our information retrieval experience tells us that common words, which occur frequently in almost every story, seem to have too much influence on the scores in Equation (3). In a vector space system, an Inverse Document Frequency (IDF) factor would account for this. We chose to approximate this effect using inverse collection frequency, revising Equation (3) by dividing by probability of the word in the background model:

$$SC'(w|T) = \frac{\lambda P(w|T) + (1-\lambda)P(w|B)}{P(w|B)} \tag{5}$$

Note that we diverge at this point from a strict language modeling framework since Equation (5) no longer represents a probability. One problem with this formulation is that the word w might not appear in the background model. To prevent the denominator $P(w|B)$ from being zero, we added the story S into the background corpus:

$$SC(w|T) = \frac{\lambda P(w|T) + (1-\lambda)P(w|B+S)}{P(w|B+S)} \tag{6}$$

Substituting $SC(w|T)$ for $P(w|T)$ in Equation (2) and ranking using the sum of the logarithms produces a score for each story S given the topic T. We also add 1 to each $SC(w|T)$ before projecting $SC(w|T)$ into logarithms so that the final scores are positive. We found during development testing, however, that long stories tended to receive higher scores than short stories. To remove this effect, we normalized the score for each story by dividing by the number of terms that the story contained. Therefore, the final score function that we used in our experiments was:

$$SC_f(S|T) = \frac{(\sum_{w \in S} log(SC(w|T)+1)) \cdot freq(w \ in \ S)}{\sum_{w_i \in S} freq(w_i \ in \ S)} \tag{7}$$

where $SC(w|T)$ is calculated as in Equation (6).

2.2. Score Normalization and Threshold Selection

Equation (7) produces scores that are comparable within a topic and a source, but a requirement for score normalization arises from two factors. First, some topic models produce higher scores than others for equally good documents. This is a natural consequence of the fact that some topics are defined using more selective terms, which (because of our approximation to inverse document frequency) receive higher scores. Second, stories from different sources may systematically use language in different ways. For example, the New York Times tends to make greater use of highly specific terms than many of the other available sources. This factor is particularly important for speech recognition transcripts (where some terms might not be correctly recognized) or for non-English text (where some terms might not be correctly translated).

We adopted a variant of the "z-score" normalization method proposed by Leek et al [7]. The method assumes that the scores of off-topic stories have a roughly Gaussian distribution, thereby making it possible to replace each score with a measure of the degree of surprise associated with each score, expressed as a distance from the mean, measured in terms of standard deviation:

$$SC_N(S,T) = \frac{SC_f(S,T) - \mu_{off}}{\sigma_{off}} \tag{8}$$

If a large number of off-topic stories are available, the sample mean μ_{off} and standard deviation σ_{off} should be reliable estimates. The difference between our approach and that reported by Leek et al was that we selected the (hopefully) off-topic stories from a different collection. For development testing on the TDT-3 collection, we used the TDT-2 collection as our source of off-topic stories. For our official runs on the TDT-4 collection, we used the TDT-3 collection for this purpose. We computed this normalization separately for each source, substituting the normalization factor for the most similar source in the case of sources that were not present in the training collection. As suggested in [7], we reported any story that received a score SC_f greater than three standard deviations above the mean as an on-topic story.

2.3. Translingual Techniques

Translingual detection is quite important with the TDT-4 collection because approximately three quarters of the stories are in languages other than English, with about half the total collection being in Arabic. We chose, therefore, to focus on

Arabic this year, using dictionary-based techniques to translate Arabic into English. For Mandarin, we used the standard translations that are provided with the collection. We have demonstrated effective techniques for Mandarin in previous years [8, 9], and we plan to integrate those techniques in our language modeling framework in the near future.

We started our Arabic processing by using Darwish's "morph" tool [5] to accomplish two functions in order: (1) transliterate Arabic orthography into ASCII letters in a way that is compatible with our downstream processing, (2) normalize Arabic characters that are often interchanged by authors to a single standard representation for each confusable set. We then found the stem for each Arabic term using the Al-stem stemmer that was provided as a standard resource for the TREC-2002 CLIR track.[1] This stemmer, developed through collaboration between the University of Maryland and the University of Massachusetts, uses one stage of hand-built rules to remove common prefixes and suffixes. Finally, we removed all Arabic stems found on a 127-entry stopword list.

We then looked up each Arabic stem in the Arabic-to-English translation probability table that was provided by BBN as a standard resource for the TREC-2002 CLIR track. For each Arabic stem that was found in the table, we replaced the occurrence of the stem with one occurrence each for the two most likely English stems. The translation probability table contains English translations for 261,971 Arabic stems, with probabilities learned from translation-equivalent UN documents using the Giza++ implementation of the IBM model 1 statistical machine translation technique [1, 4]. We refer to this translation probability table as the "statistical lexicon." The English stems contained in this table were found using the Porter stemmer [12], so we also applied the same stemmer to all terms found in English stories and in the English translations of Mandarin stories. We then removed any English stems found in a 571-entry stopword list. The remaining English terms were used.

We also performed a simple form of transliteration for each untranslated Mandarin character found in the provided translations, replacing the character with a single instance of its Pinyin representation in the hopes of matching Chinese proper names that were rendered using Pinyin in English.

3. System Implementation

Both of the systems that we built relied in part on the Lemur toolkit developed by the University of Massachusetts and Carnegie Mellon University [14] and in part on Perl code that we developed specifically for this evaluation. The systems are distinguished by how the score calculations were performed (in the Perl system, in Perl; in the Lemur system, using Lemur).

3.1. The Perl System (UMD3)

Lemur is a very flexible system, but that flexibility resulted in a fairly steep learning curve as we sorted out how best to employ the available capabilities. We were able to start working seriously with Lemur only one month before the due date for submissions, and when the due date arrived we were still looking at results on our TDT-3 development collection that were not much better than random selection. We therefore initiated a parallel development effort with a goal of clarifying our understanding of some key implementation issues, and as an insurance policy in case our work with the full Lemur system did not come together quickly. By this point, we already had high confidence in our ability to index the collection using Lemur, so we built our new system on top of Lemur's index. Thanks to excellent support from the University of Massachusetts, we had access to a version of Lemur that could perform incremental indexing. We used this for initial proof of concept but in the interest of time we ultimately chose to build a single fixed index for each collection. We chose to code the new system in Perl because its extensive string processing facilities supported rapid prototype development well.

Our Perl system consists of three subparts. The first part processes boundary files to generate token files with inline story boundaries marked. The resulting token files are then indexed using the Lemur API. This task was performed individually for the TDT-2, TDT-3, and TDT-4 collections. The second part performs tracking, first loading a background corpus (e.g., TDT-2) and a topic representation (e.g., one on-topic story from TDT-3) into memory using the Lemur API, and then computing a score for each story in the evaluation set (in this case, TDT-3). The third part then performs score normalization and applies the detection threshold as described above.

We used our Perl system to choose a value for λ in Equation (6), using the TDT-2 collection to build the background model and using TDT-3 as a development test collection. We tried values of 0.5, 0.7 and 0.85, finding 0.85 to be the best. We later discovered that we had inadvertently used only binary values for the numerator of Equation (4) when modeling topics in our Perl system (0 for absent, 1 for present—regardless of how many times). We later reran a grid of values for λ with this error corrected and found that 0.15 was actually the optimal value on the TDT-3 collection (see table 1). All of our results on the TDT-4 collection were, however, submitted with $\lambda = 0.85$ (and the Perl system results that we submitted are from the system with the binary-valued numerator error).

λ	TW Min DET Norm(Cost)
0.99	0.2012
0.85	0.1977
0.5	0.1892
0.15	0.1858
0.10	0.1908

Table 1: The effect of λ on tracking effectiveness (TDT-3, manual transcripts, reference boundaries, 1 on-topic English training story).

[1] Standard TREC CLIR track resources are available at http://www.glue.umd.edu/ dlrg/clir/trec2002/.

3.2. The Lemur system (UMD2)

With our Perl reference implementation in hand, we focused on getting our Lemur implementation to work end to end. We reused the first stage of our Perl system to prepare the token files, but did the score calculation using Lemur. Each term was scored independently using Lemur language model objects and unigram counters, and the result was combined as described in Equation (7). Score normalization and detection threshold application was then performed as in the Perl system. In addition to correcting the binary-valued numerator error, our Lemur system also proved to be considerably faster than our Perl system. The most important benefit of the Lemur implementation is the future flexibility that it provides—for example, we should be able to easily look at the effects of different smoothing techniques.

3.3. The Background Model

For our TDT-4 runs, we used a balanced combination of TDT-2 and TDT-3 collection as the background collection. We used all of the the English newswire and manually transcribed audio text in the TDT-3 collection as our representation of English documents in the background collection; about 15 million tokens (before stemming). We then added the LDC-provided English translations of the Mandarin documents in both TDT-2 and TDT-3 collections as our representation of Mandarin-to-English translations; an additional 16 million tokens (before stemming). Our only source of Arabic-to-English translations for the background model was the Arabic stories in the TDT-3 collection, a mere 3 million tokens (but after stemming). To avoid drowning those statistics in the others, we replicated every token in the Arabic-to-English translations twice (for a total of three instances) before adding them to the background collection. This results in about 9 million tokens from that source.

4. Results

We finally managed to submit a set of six runs on October 17, sixteen days after the due date. This was after the adjudication pools had been formed, so our runs are unadjudicated. Two runs were performed using the Lemur system , (*lemur-man-ldc* for the required condition (newswire+manual transcription, 1 on-topic English training story, 0 off-topic training stories, reference boundaries, and LDC translations) and (*lemur-man-our*) under the same conditions except for the use of our own translations from Arabic. The remaining four runs were performed using our Perl system. Two of these runs (*perl-man-ldc* and *perl-man-our*) paralleled to the Lemur runs. For the other two, we ran the same conditions except that automatic speech recognition transcripts were used rather than manually prepared transcripts, and 4 training stories were used (*perl-asr-ldc* and *perl-asr-our*). The *perl-asr-ldc* run is one part of the challenge condition. We did not run the other part of the challenge condition (adding 2 guaranteed off-topic training stories) because we can not presently make good use that information in our model.

As Figure 1 illustrates, our Lemur system (UMD2—the dark line marked with the square) did achieve results comparable to those of other topic tracking systems on the required condition despite our suboptimal tuning of the λ parameter.

We attribute this credible performance to a combination of a reasonable approach to language modeling and the effectiveness of z-score normalization. We attribute the relatively poor performance of our Perl system (UMD3—the isolated line marked with an "X") to our unintended use of binary term weights. Another possible factor is that we used fewer (hopefully) off-topic stories to compute the mean and standard deviation for score normalization in our Perl system (about 2,000 vs. about 7,000 in our faster Lemur system), and this may have produced less reliable estimates.

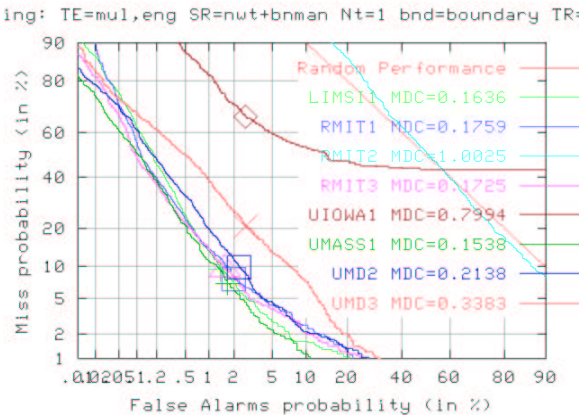

Figure 1: Required condition.

As Figure 2 shows, with our Lemur system, two-best document translation from Arabic into English using translation probabilities learned from a parallel corpus produced no improvement over the use of the Arabic-to-English translations provided by LDC. We attribute this to our use of a single source of translation knowledge in these experiments.

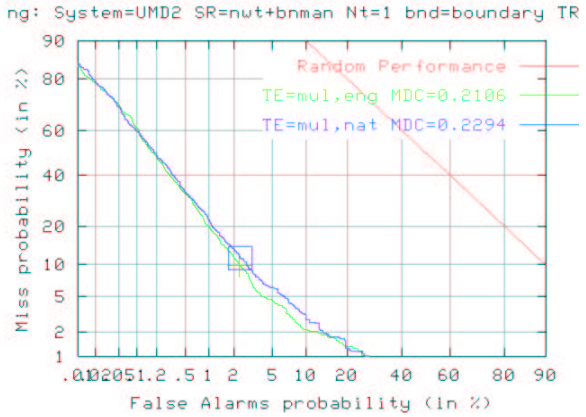

Figure 2: Lemur system, LDC Arabic translations (lighter line) vs. statistical lexicon (darker line).

Surprisingly, Figure 3 shows that our Perl system did improve substantially when using the same set of translation probabilities. At present we have no explanation for this observed effect. As expected, further improvement results when three

more on-topic training stories are available (see Figure 3).

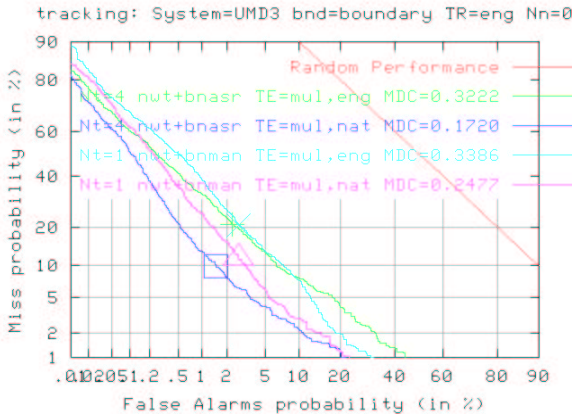

Figure 3: Topic tracking effectiveness, Perl system, LDC Arabic translations ("X") vs. statistical lexicon (triangle), with further improvement from 4 on-topic training stories (square).

5. Conclusion

We have realized our two key objectives: we built a credible topic tracking system that achieved performance comparable to that of other systems, and we did so using the Lemur toolkit. Along the way, we learned a bit about language modeling, and we developed an implementation that should serve as a useful point of reference for future work. There are several interesting directions in which this work could lead, including:

Translingual detection. For TDT-2002, we used a very simple approach to map Arabic into English, and we relied on the Chinese-to-English translations that were provided with the collection by LDC. For TDT-2003 we plan to incorporate the best results from our Arabic-English TREC-2002 CLIR system and from our earlier work on Mandarin-English translingual topic tracking. Given the importance of proper names to detection in news stories [6], we are also interested in exploring the effect of more sophisticated transliteration techniques than we tried this year.

Category models. This year we built our topic models using linear interpolation between terms found in on-topic training stories and terms found in a large background model. We also did some exploratory work with terms found in all known on-topic stories within the 11 categories LDC had annotated in the TDT-2 collection, but obtained no improvement (on the TDT-3 collection). Successful use of category models requires that we recognize the category (or categories) of a newly arrived story correctly, and that we use the corresponding category statistics appropriately. Our preliminary failure analysis indicates that both stages still need work.

Improved background models. Ideally, we would like to build the background model from the largest possible set of representative documents. We now have some experience working with the Internet Archive [2], which is able to support date-limited retrieval. This offers access to a potentially enormous source of statistics for constructing background models that would comply with TDT limitations on the allowable training epoch. The challenge will be to efficiently select a subset of the available data that is representative of TDT collection characteristics.

Unsupervised adaptation. At present, both the topic and the background model remain fixed across all newly arrived stories. We tried unsupervised adaptation of the topic model (treating newly arrived stories that received very high scores as likely to be on topic and rebuilding the topic model), but we were unable to obtain any improvement on the TDT-3 collection. The newest release of Lemur incorporates incremental indexing, which should facilitate experimentation with variants of this technique that include updates to the background model.

Alternative language models. The relatively short time between the formation of our team and the submission of our results forced an early convergence on a single approach. We are interested in exploring both structural alternatives (e.g., different ways of accommodating stories of differing lengths) and ways of exploiting additional linguistic knowledge (e.g., differential treatment of named entities).

The TDT evaluations fill a critical niche, providing the only continuing venue for evaluating detection of spoken language materials and for evaluating Arabic-English translingual detection. We look forward to continuing our exploration of these important issues over the next year.

Acknowledgments

The authors are grateful to Lise Getoor and Philip Resnik for their helpful advice throughout the work reported in this paper. This work has been supported in part by DARPA cooperative agreement N660010028910.

References

1. Y. Al-Onaizan, J. Curin, M. Jahr, K. Knight, J. Lafferty, D. Melamed, F.-J. Och, D. Purdy, N. A. Smith, and D. Yarowsky. Statistical machine translation, final report, jhu workshop 1999. Technical report, CLSP/JHU, 1999.
2. Internet Archive. http://www.archive.org.
3. A. Berger and J. Lefferty. Information retrieval as statistical translation. In *Proceedings of SIGIR'99*, pages 222–229, 1999.
4. P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

5. Kareem Darwish. Building a Shallow Morphological Analyzer in One Day. In *Proceedings of ACL2002 workshop on Computational Approaches to Semitic Languages*, 2002.

6. Dina Demner-Fushman and Douglas W. Oard. The effect of bilingual term list size on dictionary-based cross-language information retrieval. In *Thirty-Sixth Hawaii International Conference on System Sciences*, 2003. to appear.

7. Tim Leek, Richard Schwartz, and Srinivasa Sista. Probabilistic approaches to topic detection and tracking. In James Allan, editor, *Topic Detection and Tracking: Event-based Information Organization*, chapter 4, pages 67–84. Kluwer Academic Publishers, 2002.

8. Gina-Anne Levow and Douglas W. Oard. Translingual topic tracking: Applying lessons from the mei project. In *Proceedings of the TDT workshop 2000*, Gaithersburg, MD, November 2000.

9. Gina-Anne Levow and Douglas W. Oard. Signal boosting for translingual topic tracking. In James Allan, editor, *Topic Detection and Tracking: Event-based Information Organization*, chapter 9, pages 175–196. Kluwer Academic Publishers, 2002.

10. David R. H. Miller, Tim Leek, and Richard M. Schwarts. A Hidden Markov Model Information Retrieval System. In *Proceedings of SIGIR'99*, pages 214–221, 1999.

11. Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR'98*, pages 275–281, 1998.

12. Martin Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

13. Martijn Spitters and Wessel Kraaij. A Language Modeling Approach to Tracking News Events. In *Proceedings of TDT workshop 2000*, pages 101–106, 2000.

14. The Lemur toolkit. http://www.cs.cum.edu/~lemur.