TEXT SEGMENTATION AND TOPIC TRACKING ON BROADCAST NEWS VIA A HIDDEN MARKOV MODEL APPROACH

P. van Mulbregt, I. Carp, L. Gillick, S. Lowe and J. Yamron

Dragon Systems, Inc.
320 Nevada Street
Newton, MA 02460

ABSTRACT

Expertise in the automatic transcription of broadcast speech has progressed to the point of being able to use the resulting transcripts for information retrieval purposes. In this paper, we first describe a corpus of automatically recognized broadcast news, a method for segmenting the broadcast into stories, and finally apply this method to retrieve stories relating to a specific topic. The method is based on Hidden Markov Models and is in analogy with the usual implementation of HMMs in speech recognition.

1. INTRODUCTION

In [1], we introduced a new approach to text segmentation and topic tracking, one based on HMMs and classical language modeling. In that paper we applied the method to segment text from the Topic Detection and Tracking (TDT) Pilot Study Corpus, made up of Reuters newswire and manually transcribed CNN news stories. Other approaches to the problem have used loglinear models [2] or techniques from information retrieval [3, 4, 5].

Dragon Systems has now automatically transcribed approximately 800 hours of broadcast news for the TDT2 corpus. In this paper we apply the earlier work to the errorful text resulting from this automatic transcription, and attempt to take advantage of some of the features of the broadcast transcript.

2. THE TDT2 CORPUS

The TDT2 Corpus consists of about 60,000 news stories from television, radio, and newswire, collected by the Linguistic Data Consortium (LDC) over the period January 1998 through June 1998. The broadcast portion includes both closed-caption and automatic transcriptions of entire shows from the Cable News Network (CNN), American Broadcasting Company (ABC), Public Radio International (PRI), and Voice of America (VOA). The newswire component of the corpus was collected from the Associated Press Worldstream (APW) and the New York Times News Service(NYT).

The automatic transcriptions of the broadcast portion of the corpus include time-aligned output, together with a measure of the recognizer's confidence and a speaker cluster identifier, for each word. They were provided by Dragon Systems using a speech recognizer essentially similar to the Dragon 1997 Hub4 system, which recorded a word error rate (WER) of 23.4% in that evaluation [6, 7]. Its performance on the TDT2 data, which is somewhat more difficult, is about 30% WER. The recognition was done on commercially available hardware, and ran about 15 times slower than real time on a 300MHz machine.

Story boundary determination for the automatically transcribed broadcast was done by taking the boundaries in the closed caption data, recording their times relative to the start of the program, and then finding the word in the recognizer output which was closest in time and marking a story boundary at that location.

To facilitate the tracking evaluation, every story in the corpus was manually labeled as on or off topic for each topic in a set of 100, selected arbitrarily from the time frame of the corpus. Stories were permitted to be on multiple topics. A topic was defined to be "a seminal event or activity, along with all directly related events and activities"[8].

3. THE SEGMENTER

Suppose that there are k topics $T^{(1)}$, $T^{(2)}$, $\ldots$, $T^{(k)}$. There is a language model associated with each topic $T^{(i)}$, $1 \leq i \leq k$, with which one can calculate the probability of any sequence of words. In addition, there are transition probabilities among the topics, including a probability for each topic to transition to itself (the "self-loop" probability), which implicitly specifies an expected duration for that topic. Given a text stream, a probability can be attached to any particular hypothesis about the sequence and segmentation of topics in the following way:

1. Transition from the start state to the first topic and accumulate a transition probability.
2. Stay in topic for a certain number of words or sentences, and, given the current topic, accumulate a self-loop probability and a language model probability for each.
3. Transition to a new topic, accumulate the transition probability, and go back to step 2.

A search for the best hypothesis and corresponding segmentation can be done using standard HMM and speech recognition techniques. This segmentation does provide a label for each segment, namely the topic to which that segment was assigned in the best path, but the label may not have much meaning to a human.

3.1. Constructing the Topic Models

The topic language models used by the segmenter were built from the newswire and the automatically transcribed broadcasts from the January and February data. This totaled about 6 million words spread across 14586 stories, though the average number of words per story varied from a low of 129 for CNN, to a high of 850 for the New York Times. A global unigram model consisting of 60,000 unique words was built from this data.

Topic clusters were constructed by automatically clustering the stories in the training data. This clustering was done using a multi-pass k-means algorithm described in [1]. In order to prevent very common words and punctuation symbols from dominating the computation, we introduced a stop list containing 112 entries. These words did not participate in the computation of the distance measure. Removing these words from the vocabulary meant that approximately half the words in the text stream were not scored.

A topic language model was built from each cluster. To simplify this task, we limited the number of clusters to 100 and chose to model each topic using unigram statistics only. These unigram models were smoothed versions of the raw unigram models generated from the clusters. Smoothing each model consisted of performing absolute discounting followed by backoff [9] to the global unigram model; in other words, a small fixed count (about .5) was subtracted from the non-zero raw frequencies, and the liberated counts were redistributed to the rest of the words in the model in proportion to the global unigram distribution built from the training data. The raw cluster unigrams were quite sparse, typically containing occurrences of only 6,000 distinct words from the training list of 60,000 words. Words on the stop list were removed from the models.

We will frequently refer to these topic language models as *background topics* or *background models.*

3.2. Segmentation Results

Experiments were performed on the automatically transcribed portion (ASR) of the TDT2 Corpus, from the months of March and April. This collection comprised about 350 shows, 6000 stories, 2 million words. About 75% of the shows are from CNN. The ASR output has breaks marked – typically silence, music or speech with a music background – and were used to identify possible story transition times.

Decoding of text was done by using a speech recognizer with 100 underlying "single node" models (corresponding to the topics), each of which was represented by a unigram model as described above. The text was scored against these models one *frame* at a time—a frame corresponding, in these experiments, to the words between recognizer breaks. The topic-topic transition penalties were folded into a single number, the topic-switch penalty, which was imposed whenever the topic changed between segments.

The topic-switch penalty was tuned to produce the correct average number of words per segment on the test set. There are no other parameters to tune except the search beam width, which was set large enough to avoid search errors in our experiments.

The segmenter was first run on the ASR portion of the TDT2 Corpus. Then the segmenter was run on the newswire portion of the corpus, the newswire stories having been collected chronologically into similar length pieces as the broadcast shows. This latter experiment was included to test the robustness of the system to a mismatch between the training material for the background models, which was all from broadcast transcripts, and the test material.

3.2.1. TDT2 Corpus

On the ASR portion of the TDT2 corpus, the segmenter hypothesized 10437 segment boundaries compared to the 10052 story boundaries in the test set. The number of exact matches was 3799, a recall rate of 37.8%,and a precision of 36.4%. Table 1 shows the results broken out by source. Many times the hypothesized boundary missed the correct boundary by only a few words. The recall rate rises to 52.3% if we consider a boundary within 10% of the average story length a match. This performance is not as good as in [1], possibly due to the much shorter story length of the ASR text in TDT2 as compared to TDT1.

Models	CNN	ABC	PRI
Recall/Precision Exact	40.2/37.8	32.3/31.5	25.4/34.5
Recall/Precision 10%	51.7/48.6	56.5/49.6	52.0/58.0

Table 1: Segmentation Performance on ASR data broken out by source.

The types of errors made by the segmenter fall into the following categories:

- Failure to distinguish a boundary between successive stories because they were assigned to the same background topic. This didn't seem to be a large source of errors, but the effect can be reduced by increasing the number of background models.
- Failure to accurately position boundaries relative to "broadcast filler", such as, "More news after this." This is a weakness of a system that does not model story structure.
- Splitting of stories at internal topic shifts. This "problem" actually goes to the heart of what it is we are trying to accomplish, and it is not clear that this is always undesirable behavior.
- Oscillation of topic in stories not well-modeled by the background topics. This might be solved by using models with better discrimination, such as bigram models, or models that adaptively train so as to stay current.
- Oscillation of topic in long stories. In some cases knowledge of the structures of the show (some initial one sentence headlines, a very long story somewhere near the middle of the show), could be used to make a better determination of the boundaries.
- A portion of the story boundaries do not occur at breaks as output by the automatic speech recognizer, and as such can never be matched exactly.

3.2.2. Newswire vs ASR

In order to determine the robustness of our algorithm and training, the segmenter was given the (artificial) task of segmenting newswire text. Stories in the APW data were only about 40% as long as those in the NYT data, 350 words vs 850 words. The segmenter is choosing boundaries by finding regions which are semantically coherent, and it may be the case that the longer stories of the NYT give the segmenter more material to work with.

Models	NYT	APW
Recall/Precision	63.4/58.0	37.2/33.8

Table 2: Segmentation Performance on Newswire data broken out by source.

3.2.3. Immediate Decision

The approach we have taken requires the whole text stream to be processed first. Then a traceback occurs and the boundaries determined by the segmentation are marked. It is possible to restrict the segmenter so that at any position in the text stream, it only looks ahead a fixed number of words and must output a decision immediately as to whether or not a segment boundary occurs at this position. This capability might be desired for an "online system". Experiments performed on the TDT2 corpus suggest that there is a small degradation in the method as the length of the look ahead period decreases, but that if the segmenter is allowed to look ahead twice the average story length, then it almost never changes its mind.

3.2.4. Varying the Number of Background Models

The choice of 100 background models was somewhat arbitrary. To see how the system performed with more models, and to try to fix the first problem above, models were built after clustering the data into 250 clusters. While this does provide more resolution, it also results in clusters with less data and perhaps less robust estimates for the unigram probabilities.

Models	CNN	ABC	PRI
Recall/Precision	45.5/33.2	35.2/28.8	31.3/30.0

Table 3: Segmentation Performance on ASR data broken out by source for models with 250 clusters.

3.2.5. Using the Confidence Measure

The ASR output is provided with a confidence measure, a number between 0 and 1 identifying how confident the recognizer is in its choice of that particular word at that time. To investigate the use of confidence, we tried raising the probability of a word given a topic to the confidence, or equivalently multiplying the log probability by the confidence. This has the affect of downplaying difference in scores between two language models if the confidence of the word is close to 0. In segmentation (and tracking) experiments, no significant difference was found in the results obtained in this manner.

A histogram of confidences for correct words shows that the recognizer is reasonably sure of itself – the bin height is correlated with the confidence value. On the other hand, the confidence for words that are misrecognized have a rather flat distribution. This may mean that the contribution from misrecognized words is not easy to discount.

Another observation was that average confidence value across all words is around 70%. After removing words in the stop list, the remaining words have an average confidence closer to 80%. It may be that the words in the story with information are being given higher confidences with less variation and so their effect is not being diminished.

4. THE TRACKER

The topic tracker is an adaptation of the segmenter. As discussed above, the segmentation algorithm does segmentation and topic assignment simultaneously. In general, the topic labels assigned by the segmenter (which are drawn from the set of automatically derived background topics) are not useful for classification, as they are few in number and do not necessarily correspond to categories a person would find interesting. However, by supplementing the background topic models with a language model for a specific topic of interest, and allowing the segmenter to score segments against this model, it becomes possible for the segmenter to output a notification of an occurrence of that topic in the news stream whenever it assigns that topic model's label to a story. In this implementation, the background models have the role of determining the background against which the topic model must score sufficiently well to be identified.

In this incarnation, the segmenter is not asked to identify story boundaries, instead using known boundaries. Its job is merely to score each story against its set of background models, as well as against the topic model, and report the score difference between the best background model and the topic model. A threshold is applied to this difference to determine whether a story is about the topic or not, and this threshold can be adjusted to tune the tradeoff between missing and falsely reporting stories on the topic.

4.1. Constructing the Topic Models

A topic model is built from N_t training stories, where N_t is some fixed value not greater than 16. As language models, these are extremely sparse and must be smoothed, which we do in a manner similar to the background models:

1. Apply our stop list of 112 common words and punctuation, so that they don't participate in the topic determination.
2. Steal a small amount (about .5 count) from the non-zero raw frequencies using absolute discounting.
3. Redistribute the liberated counts to the rest of the words in the topic model in proportion to their occurrence in a backoff unigram distribution.

In this case, in order to provide a more accurate smoothing for the topic model, the backoff unigram distribution is not the one generated from the segmentation training data. Instead, we take as the backoff distribution the mixture of the background topic models that best approximates the unsmoothed topic model. There is therefore a different backoff model for every topic and every value of N_t.

4.2. Tracking Results

A list of 100 topics has been defined by the LDC and all stories in the corpus are marked as either on or off topic for that topic. A story is allowed to be on multiple topics. These topics include the "Asian Financial Crisis", "Indian Parliamentary Elections", "U.S. National Tobacco Settlement", "March 1998 Asteroid scare". Some had many stories, but others, such as the "FDA approval of Viagra" had only a few in the period under consideration. For the period March and April 1998, there were 8 topics with 16 or more stories.

The tracking results comprise a large number of experiments, each performed as follows:

1. Choose a topic E and a number of training examples N_t
2. Find the first 16 stories labeled as topic E in the corpus.
3. Train a language model for topic E (as described above) from the last N_t stories with label E prior to and including the last story identified in step 2.
4. Run the tracker on all stories after the 16^{th} story labeled with topic E. (Testing on all stories after the 16^{th} labeled story makes the test set the same for all values of N_t.)

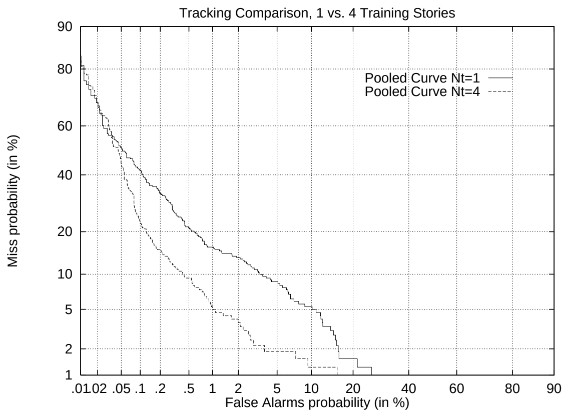

Figure 1: Tracking performance for different values of $N_t = 1, 4$ on TDT2

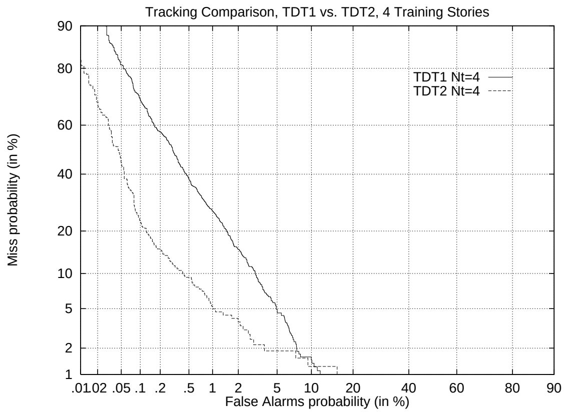

Figure 2: Tracking performance comparing TDT1 and TDT2, with 4 training stories

This procedure was repeated for all possible values of E and N_t.

The only tunable parameter in this system is the threshold on the topic-background score difference. This threshold was varied to produce full plots of miss *vs.* false alarm.

Figure 1 shows the results of our tracking experiments for $N_t = 1$ and $N_t = 4$, averaged over all events. The results for four training stories are substantially better than those for one training story, though the two plots do come together at low false alarm rates.

Figure 2 shows a comparison between TDT1 and TDT2 data, both using 4 training stories. It appears from this graph that the TDT2 test data is somewhat easier than the TDT1 data. The TDT2 definition of topic is somewhat broader than that of TDT1, but that wouldn't be expected to make the task any easier.

The smoothing of the event models has been improved, resulting in a sharp improvement in performance at small numbers of training stories compared to the results reported in [1]. However, the time penalty used there was not turned on for these experiments; activating it would improve performance at high values of the threshold (high miss, low false alarm).

5. THE FUTURE

It is possible to improve the topic modeling for the segmenter. Adding language models for the beginning and end of a story will address one of the types of error made by the segmenter.

For the tracking task, one key is to be able to adapt to the topic as it evolves over time. New words get added to a topic (the identity of the Oklahoma bomber, Timothy McVeigh) and other words get dropped (the 1994 earthquake in Kobe, quickly became the "earthquake in Japan", and references to Kobe were dropped). Adapting the event model, in an unsupervised manner, on stories that the tracker believes are on topic offers an opportunity to improve performance.

We believe we are having problems in properly classifying very small stories. In particular, these may be leading to false alarms at high threshold.

Instead of using the story boundaries provided by the human labelers, it is natural to investifate the effect of using the story boundaries provided by the segmenter on topic tracking.

Dragon looks forward to implementing these ideas in a future system.

6. REFERENCES

[1] J.P. Yamron, I. Carp, L. Gillick, S. Lowe, and P. van Mulbregt, "A Hidden Markov Model Approach to Text Segmentation and Event Tracking," *Proceedings ICASSP-98, Seattle, May 1998*

[2] D. Beeferman, A. Berger, and J. Lafferty, "Text segmentation using exponential models," in *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing,* Providence, RI, 1997.

[3] M.A. Hearst, "Multi-paragraph Segmentation of Expository Text," in *Proceedings of the ACL,* 1994.

[4] H. Kozima, "Text Segmentation Based on Similarity between Words," in *Proceedings of the ACL,* 1993.

[5] J.M. Ponte and W.B. Croft, "Text Segmentation by Topic," in *Proceedings of the First European Conference on Research and Advanced Technology for Digitial Libraries,* pp. 120–129, 1997.

[6] Steven Wegmann, Francesco Scattone, Ira Carp, Larry Gillick, Robert Roth, Jonathan P. Yamron., "Dragon Systems' 1997 Broadcast News Transcription System,"' *Proceedings of Broadcast News Transcription and Understanding Workshop,* Lansdowne, Virginia, Feb 1998.

[7] L. Gillick, Yoshiko Ito, Linda Manganaro, Michael Newman, Francesco Scattone, S. Wegmann, Jon Yamron, Puming Zhan, " Dragon Systems' Automatic Transcription of New TDT Corpus," *Proceedings of Broadcast News Transcription and Understanding Workshop,* Lansdowne, Virginia, Feb 1998.

[8] "The Topic Detection and Tracking Phase 2 (TDT2) Evaluation Plan" available from <http://www.nist.gov/speech/tdt98/tdt98.htm>

[9] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," in *IEEE Transactions on Acoustics, Speech and Signal Processing,* ASSP-35(3):400–401, March, 1987.