Topic Detection and Tracking using idf-Weighted Cosine Coefficient

J. Michael Schultz, Mark Liberman

University of Pennsylvania
Philadelphia, PA, 19104

ABSTRACT

The goal of TDT Topic Detection and Tracking is to develop automatic methods of identifying topically related stories within a stream of news media. We describe approaches for both detection and tracking based on the well-known idf-weighted cosine coefficient similarity metric. The surprising outcome of this research is that we achieved very competitive results for tracking using a very simple method of feature selection, without word stemming and without a score normalization scheme. The detection task results were not as encouraging though we attribute this more to the clustering algorithm than the underlying similarity metric.

1. The Tracking Task

The goal of the topic tracking task for TDT2 is to identify news stories on a particular event defined by a small number (Nt) of positive training examples and a greater number of negative examples. All stories in the news stream subsequent to the final positive example are to be classified as on-topic if they pertain to the event or off-topic if they do not. Although the task is similar to IR routing and filtering tasks, the definition of event leads to at least one significant difference. An event is defined as an occurrence at a given place and time covered by the news media. Stories are on-topic if they cover the event itself or any outcome (strictly-defined in [2]) of the event. By this definition, all stories prior to the occurrence are off-topic, which contrary to the IR tasks mentioned, theoretically provides for unlimited off-topic training material (assuming retrospective corpora are available). We expected to be able to take advantage of these unlimited negative examples but in our final implementation did so only to the extent that we used a retrospective corpus to improve term statistics of our database.

1.1. idf-Weighted Cosine Coefficient

As the basis for our approach we used the idf-weighted cosine coefficient described in [1] often referred to as $tf \cdot idf$. Using this metric, the tracking task becomes two-fold. Firstly, choosing an optimal set of features to represent topics, i.e. feature selection. The approach must choose features from a single story as well as from multiple stories (for $Nt > 1$). Secondly, determining a threshold (potentially one per topic) which optimizes the miss and false alarm probabilities for a particular cost function, effectively normalizing the similarity scores across topics.

The cosine coefficient is a document similarity metric which has been investigated extensively. Here documents (and queries) are represented as vectors in an n-dimensional space, where n is the number of unique terms in the database. The coefficients of the vector for a given document are the term frequencies (tf) for that dimension. The resulting vectors are extremely sparse and typically high frequency words (mostly closed class) are ignored. The cosine of the angle between two vectors is an indication of vector similarity and is equal to the dot-product of the vectors normalized by the product of the vector lengths.

$$\cos(\theta) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \|\vec{B}\|}$$

$tf \cdot idf$ (term frequency times inverse document frequency) weighting is an ad-hoc modification to the cosine coefficient calculation which weights words according to their *usefulness* in discriminating documents. Words that appear in few documents are more useful than words that appear in many documents. This is captured in the equation for the inverse document frequency of a word:

$$idf(w) = \log_{10}\left(\frac{N}{df(w)}\right)$$

Where $df(w)$ is the number of documents in a collection which contain word w and N is the total number of documents in the collection.

For our implementation we weighted only the topic vector by idf and left the story vector under test unchanged. This allows us to calculate and fix an idf-scaled topic vector immediately after training on the last positive example story for a topic. The resulting calculation for the similarity measure becomes:

$$sim(a,b) = \frac{\sum_{w=1}^{n} tf_a(w) \cdot tf_b(w) \cdot idf(w)}{\sqrt{\sum_{w=1}^{n} tf_a^2(w)} \cdot \sqrt{\sum_{w=1}^{n} tf_b^2(w)}}$$

1.2. UPENN System Attributes

To facilitate testing, the stories were loaded into a simple document processing system. Once in the system, stories are processed in chronological order testing all topics simultaneously with a single pass over the data[1] at a rate of approximately 6000 stories per minute on a Pentium 266 MHz machine. The system tokenizer delimits on white space and punctuation (and discards it), collapses case, but provides no stemming. A list of 179 stop words consisting almost entirely of close classed words was also employed. In order to improve word statistics, particularly for the beginning of the test set, we prepended a retrospective corpus (the TDT Pilot Data [3]) of approximately 16 thousand stories.

[1] In accordance with the evaluation specification for this project [2] no information is shared across topics.

1.3. Feature Selection

The *choice* as well as *number* of features (words) used to represent a topic has a direct effect on the trade-off between miss and false alarm probabilities. We investigated four methods of producing lists of features sorted by their effectiveness in discriminating a topic. This then allowed us to easily vary the number of those features for the topic vectors[2].

1. Keep all features except those words belonging to the stop word list.
2. Relative to training stories, sort words by document count, keep n most frequent. This approach has the advantage of finding those words which are common across training stories, and therefore are more general to the topic area, but has the disadvantage of extending poorly from the $Nt = 16$ case to the $Nt = 1$ case.
3. For each story, sort by word count (tf), keep n most frequent. While this approach tends to ignore low count words which occur in multiple training documents, it generalizes well from the $Nt = 16$ to the $Nt = 1$ case.
4. As a variant on the previous method we tried adding to the initial n features using a simple greedy algorithm. Against a database containing all stories up to and including the Nt-th training story, we queried the database with the n features plus the next most frequent term. If the separation of on-topic and off-topic stories increased, we kept the term, if not we ignored it and tested the next term in the list. We defined separation as the difference between the average on-topic scores and the average of the 20 highest scoring off-topic documents.

Of the feature selection methods we tried the forth one yielded the best results across varying values of Nt, although only slightly better than the much simpler third method. Occam's Razor prompted us to omit this complication from the algorithm. The DET curves[3] in Figure 1. show the effect of varying the number of features (obtained from method 3) on the miss and false alarm probabilities. The upper right most curve results from choosing the single most frequent feature. Downward to the left, in order are the curves for 5, 10, 50, 150 and 300 features. After examining similar plots from the pilot, training[4], and development-test data sets, we set the number of features for our system to 50. It can be seen that there is limited benefit in adding features after this point.

1.4. Normalization / Threshold Selection

With a method of feature selection in place, a threshold for the similarity score must be determined above which stories will be deemed on-topic, and below which they will not. Since each topic is represented by its own unique vector it cannot be expected that the same threshold value will be optimal across all topics unless the scores are normalized. We tried two approaches for normalizing the topic similarity scores.

For the first approach we calculated the similarity of a random sample of several hundred off-topic documents in order to estimate an

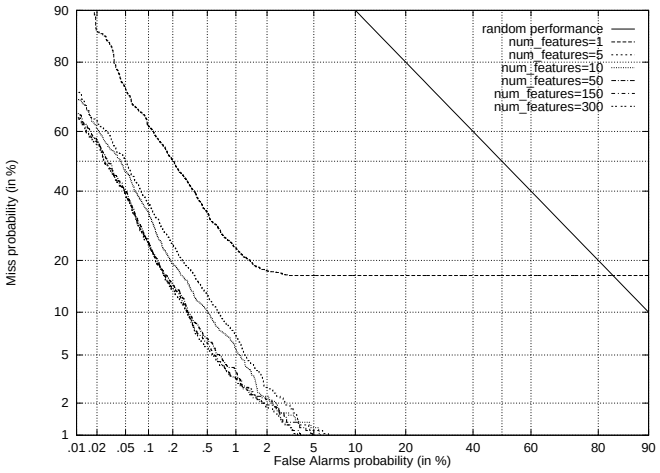

Figure 1: DET curve for varying number of features. (Nt=4, TDT2 evaluation data set, newswire and ASR transcripts)

average off-topic score relative to the topic vector. The normalized score is then a function of the average on-topic[5] and off-topic scores and the off-topic standard deviation[6]. The second approach looked at only the *highest* scoring *off-topic* stories returned from a query of the topic vector against a retrospective database with the score normalized in a similar fashion to the first approach.

Both attempts reduced the story-weighted miss probability by approximately 10 percent at low false alarm probability relative. However, this results was achieved at the expense of higher miss probability at higher false alarm probability, and a higher cost at the operating point defined by the cost function for the task[7].

$$C_{track} = C_{miss} \cdot P(miss) \cdot P_{topic} + C_{fa} \cdot P(fa) \cdot (1 - P_{topic})$$

where

$C_{miss} = 1.$ (the cost of a miss)
$C_{fa} = 1.$ (the cost of a false alarm)
$P_{topic} = 0.02.$ (the *a priori* probability of a story being on a given topic was chosen based on the TDT2 training topics and training corpus.)

Because of the less optimal trade-off between error probabilities at the point defined by the cost function, we choose to ignore normalization and look directly at cost as a function of a single threshold value across all topics. We plotted $tf \cdot idf$ score against story and topic-weighted cost for the training and development-test data sets. As our global threshold we averaged the scores at which story and topic-weighted cost were minimized. This is depicted in figure 2.

Figure 3 shows the same curves for the evaluation data set. The threshold resulting from the training and development test data applies satisfactorily though far from optimally to the evaluation data set. An optimal threshold of 39 would have improved the topic-weighted score by 17.6 percent and the story weighted cost by 1.9

[2] We did not employ feature selection on the story under test but used the text in entirety.

[3] See [5] for detailed description of DET curves.

[4] The first two month period of TDT2 data is called the training set, not to be confused with training data.

[5] calculated from the training stories.

[6] σ(on-topic) is unreliable for small Nt but for larger Nt the σ(off-topic) was found to be a good approximation of σ(on-topic).

[7] Defined in [2].

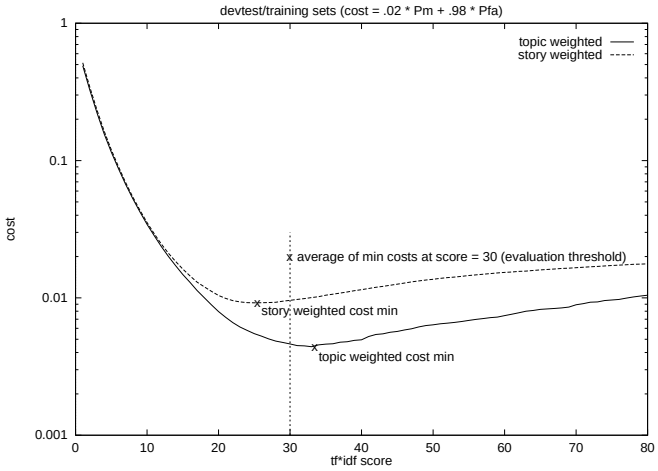

Figure 2: Story and topic-weighted cost as a function of $tf \cdot idf$ score. ($Nt = 4$, TDT2 training and development test data sets, newswire and ASR transcripts)

percent.

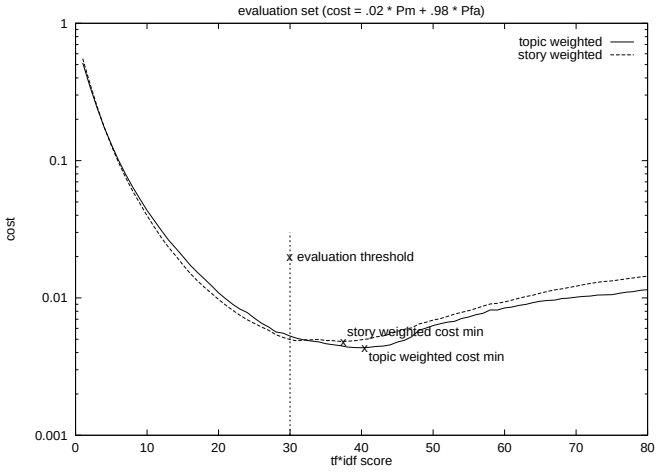

Figure 3: Story and topic-weighted cost as a function of $tf \cdot idf$ score. ($Nt = 4$, TDT2 evaluation data set, newswire and ASR transcripts)

1.5. Tracking Results and Conclusions

We tried a number of approaches to optimize the $tf \cdot idf$ weighted cosine coefficient for the tracking task. In the end very simple feature selection with no normalization of topic scores performed best and was competitive with approaches from other sites. In the present framework it appears a substantial improvement in performance would result from a better estimation of global topic threshold.

2. Topic Detection

Using the infrastructure developed for the tracking task, we implemented a simple detection system in a few weeks with the goal of becoming better acquainted with the task. The system is based on the same *idf*-weighted cosine coefficient and on single-linkage (nearest-neighbor) clustering[8].

Site	*Story Weighted*		
	$P(miss)$	$P(fa)$	C_{track}
UPenn1	0.0934	0.0040	0.0058
UMass1	0.0855	0.0043	0.0059
BBN1	0.1415	0.0035	0.0063
Dragon1	0.1408	0.0043	0.0070
CMU1	0.2105	0.0035	0.0077
GE1	0.1451	0.0191	0.0216
UMd1	0.8197	0.0062	0.0225
UIowa1	0.0819	0.0492	0.0499

Table 1: Story weighted tracking results by site. ($Nt = 4$, TDT2 evaluation data set, newswire and ASR transcripts)

Site	*Topic Weighted*		
	$P(miss)$	$P(fa)$	C_{track}
BBN1	0.1185	0.0033	0.0056
UPenn1	0.1092	0.0045	0.0066
Dragon1	0.1054	0.0049	0.0069
UMass1	0.1812	0.0038	0.0074
CMU1	0.2660	0.0023	0.0076
GE1	0.1448	0.0226	0.0251
UMd1	0.6130	0.0156	0.0275
UIowa1	0.1461	0.0425	0.0445

Table 2: Topic weighted tracking results by site. ($Nt = 4$, TDT2 evaluation data set, newswire and ASR transcripts)

The goal of the detection task is to identify stories pertaining to the same event without the aid of either positive or negative training stories. As a system parameter, a deferral period is defined to be the number of files (each containing multiple stories) the system is allowed to processes before it must associate a topic id with the stories contained in the files.

2.1. Single-Linkage Clustering

This method of agglomerative clustering begins with all stories in their own singleton clusters. Two clusters are merged if the similarity between any story of the first cluster and any story of the second cluster exceeds a threshold.

To implement the clustering, we took the stories of each deferral period and created an inverted index. Then each story, in turn, is compared with all preceding stories (including those from previous deferral periods). When the similarity metric for two stories exceeds a threshold[9] their clusters are merged. Of course, the clusters of earlier deferral periods cannot merge since the cluster id for the stories from those periods have already been reported.

[8] See [4] for a good description of single-linkage clustering.

[9] We used the best threshold based on the development test data set.

2.2. Detection Results and Conclusions

There is a major shortcoming to this approach which we chose to accept due to the ease of implementation and constraints of time. Two clusters which one would expect to remain distinct when examining their content, may become merged through intermediary stories in an effect called chaining. We found this particularly troublesome in the TDT detection task since a small variation in threshold often leads to completely different topic candidate clusters for scoring. As the threshold increases the best candidate clusters grow until a chain occurs which brings in so much off-topic material as to make it no longer the best candidate for the topic. This type of phenomenon makes incremental progress extremely difficult. We expect group-average clustering or incremental clustering to help us around this problem.

The simple single-linkage clustering algorithm performed moderately well given its inherent shortcomings. It seems clear to us a successful clustering algorithm must incorporate a representation for a cluster itself as group average clustering does, in order to avoid the problem of chaining and its resulting difficulties.

Site	*Story Weighted*		
	$P(miss)$	$P(fa)$	C_{track}
BBN1	0.0941	0.0021	0.0040
UMass1	0.0913	0.0022	0.0040
Dragon1	0.1638	0.0013	0.0045
IBM1	0.1965	0.0007	0.0046
UPenn1	0.2997	0.0011	0.0070
CMU1	0.3526	0.0004	0.0075
CIDR1	0.3861	0.0018	0.0095
UIowa1	0.6028	0.0009	0.0129

Table 3: Story weighted detection results by site. ($deferal = 10$, TDT2 evaluation data set, newswire and ASR transcripts)

Site	*Topic Weighted*		
	$P(miss)$	$P(fa)$	C_{track}
IBM1	0.1766	0.0007	0.0042
BBN1	0.1295	0.0021	0.0047
Dragon1	0.1787	0.0013	0.0048
CMU1	0.2644	0.0004	0.0057
UPenn1	0.2617	0.0011	0.0063
UMass1	0.2091	0.0023	0.0064
CIDR1	0.3309	0.0018	0.0084
UIowa1	0.4311	0.0009	0.0095

Table 4: Topic weighted detection results by site. ($deferal = 10$, TDT2 evaluation data set, newswire and ASR transcripts)

References

1. G. Salton and M.J. McGill, "Introduction to Modern Information Retrieval," *McGraw Hill Book Co.*, New York, 1983.
2. G. Doddington, "The Topic Detection and Tracking Phase 2 (TDT2) Evaluation Plan," *Available at http://www.nist.gov/speech/tdt_98.htm*, 1998.
3. G. Doddington, "The TDT Pilot Study Corpus Documentation," *Available at http://www.ldc.upenn.edu/TDT/Pilot /TDT.Study.Corpus.v1.3.ps*, 1997.
4. B. Everitt, "Cluster Analysis," *Halsted Press*, New York, 1993.
5. A. Martin, G. Doddington, T. Kamm, M. Ordowski, M. Przybocki, "The DET Curve in Assessment of Detection Task Performance," *EuroSpeech 1997 Proceedings Volume 4*, 1997.