TDT2001 Topic Tracking at RMIT University

Vaughan R. Shanks *Hugh E. Williams*

School of Computer Science and Information Technology
RMIT University
GPO Box 2476V, Melbourne 3001, Australia

ABSTRACT

This is RMIT's first year of participation in the TDT evaluation. Our system uses a linear classifier to track topics and an approach based on our previous work in document routing. We aimed this year to develop a baseline system, and to then test selected variations, including adaptive tracking. Our contribution this year to have implemented an efficient system, that is, one that maximises tracking document throughput and minimises system overheads.

1. Topic Tracking Objectives

For this year's evaluation we participated only in the topic tracking task. Given a few example stories on a particular topic, the task of a tracking system task is to process a supplied list of data files, and classify all stories in these files as either on-topic or off-topic. This task is repeated for over 100 topics.

There are few examples provided for tracking a topic. Under the TDT evaluation rules, only one, two, or four on-topic examples are provided, and only zero or two off-topic examples. To simulate field conditions for a tracking system, no data may be used that is as recent, or more recent than the data supplied. Therefore, any term statistics or pseudo-relevance feedback used must be gathered from sources published before 1998.

1.1. System Objectives

For this year's evaluation, we aimed to implement a baseline system that would produce reasonable, TDT-compliant results. We restricted our problem domain to English language documents and automatically-translated Mandarin; we did not consider any ASR texts, and only processed newswire and manually-transcribed broadcast news stories.

Our secondary goals were efficiency—a similar goal to our recent categorisation research [7]—and to experiment with various term extraction and profiling techniques to see which work better and why.

2. Implementation Details

The system we used was based on a Rocchio [6] routing categoriser we previously developed for fast document categorisation [7]. The system was developed in C and experiments were carried out on a Pentium III based server with 256 megabytes of main-memory. In this environment, our system can process around 10 gigabytes of SGML-formatted text per hour.

Terms were extracted as single words, delimited by whitespace. All terms extracted were case-folded. Words were then stemmed using Porter's algorithm [4]. We found that use of a Porter stemmer is a significant overhead: the throughput rate of our system was reduced to half when the stemmer was employed. We plan in the future to use a simpler or dictionary stemmer for efficiency. A list of 477 common English words was used as a stoplist.

As a background model of news document text, we used newswire documents from Associated Press, Wall Street Journal, and the Federal Register from TREC disks 1 and 2 [1]. Term document frequency statistics were gathered and stored to an auxiliary vocabulary file. Along with the document frequency (DF), we store the Okapi [5] inverse document frequency of these terms:

$$IDF = \ln\left(\frac{N - df + 0.5}{df + 0.5}\right)$$

In cases where $df < \frac{N}{2}$, resulting in a negative inverse document frequency (IDF), we set IDF to zero.

A document term frequency (TF) factor is calculated using an Okapi term weight:

$$TF = \frac{tf}{0.5 + 1.5 \times \left(\frac{dl}{dl_{avg}} + tf\right)}$$

where dl is the length of the document, dl_{avg} is the mean document length in the auxiliary corpus, and tf is the number of times the term occurs in the current document. This factor is multiplied by the IDF value for that term to produce a weight w:

$$w = TF.IDF$$

The weight of each term in the document is stored along with the index of that term in the dictionary to form a vector of weights. New terms observed while processing the TDT data files are added to the term dictionary with a $DF = 1$.

Topics profiles are maintained for each topic. The topic profile is calculated initially by generating weight vectors for each of the training examples supplied, then adding the positive, and subtracting the negative examples, to form one vector. Following the well-known heuristic [2, 3], negative weights in the profile are zeroed; the Rocchio parameters β and γ are therefore both set to 1.0. The resultant vector is then normalised to unit size.

The similarity sim between the topic profile vector, u, and a document vector, v, are calculated using the well-known cosine measure [8]:

$$sim = \cos(\theta) = \frac{u.v}{|u|.|v|}$$

The threshold for classification decisions is set by calculating the cosine measure of each of the training examples against the topic profile. The maximum off-topic and minimum on-topic values are recorded and a threshold calculated according to the following formula:

$$thresh = \begin{cases} \text{off}_{max} > \text{on}_{min} & : \quad \text{on}_{min} \\ \text{off}_{max} < \text{on}_{min} & : \quad \text{off}_{max} + \left(\frac{\text{on}_{min} - \text{off}_{max}}{TWB}\right) \end{cases}$$

Where TWB is a threshold weight bias, which defaults to 4, but can be set higher in order to lower the threshold and give better recall. A similarity higher than the threshold results in a YES decision for that document, while a similarity lower than the threshold results in a NO.

2.1. Adaptive Profiling

Simple, adaptive profiling was built into the system as an optional feature. With adaptive profiling enabled, the system performs unsupervised learning. Documents that are selected as on-topic—those with a YES decision—are added to the topic profile. Documents that are off-topic—those with a NO decision—are subtracted from the topic profile. A fixed number of the highest weighted terms are retained in an *active profile* which is used for performing similarity calculations. In our initial implementation described here, translated Mandarin documents were not used in adaptive feedback as it was found that the limited vocabulary in these documents results in skewing of the profile towards recognising all Mandarin documents as on-topic examples.

Document vectors that are used to adapt the profile are scaled-down to reduce the possibility of a false decision distorting the profile. A certainty factor is calculated for each decision made as follows:

$$cert = \begin{cases} \text{sim} > \text{thresh} & : \quad \text{pos}_{max} \times \frac{\text{sim} - \text{thresh}}{\text{pos}_{max} - \text{thresh}} \\ & : \\ \text{sim} < \text{thresh} & : \quad \text{neg}_{min} \times \frac{\text{thresh} - \text{sim}}{\text{thresh} - \text{neg}_{min}} \end{cases}$$

Where pos_{max} and neg_{min} are the maximum positive and minimum negative similarity values from the training examples. The document vector is normalised, then multiplied by the certainty before addition to the profile vector. The document vector can be further scaled-down using a scale factor. In the runs submitted to the TDT evaluation, we scaled these down by a factor of five; this was an arbitrary choice and we plan to experiment with this approach further in the future.

2.2. Pseudo Relevance Feedback

Optional pseudo-relevance feedback (PRF) is also included in the system, as a method to include more related terms and raise recall. Three gigabytes of TREC data were chosen as a source of PRF. PRF is introduced into the system after training is complete and the topic profile has been normalised, but before a threshold has been chosen.

The MG [8] full-text information retrieval engine (`mg-2.1`) is used to generate PRF. A ranked query is constructed from the highest-weighted three terms in the topic profile, and the top ten documents returned by a ranked query are processed into document vectors. These document vectors are then added to the topic profile in the same manner as adaptive feedback documents, except that certainty is set to the same value as similarity.

3. Experiments

Prior to the release of the TDT2001 evaluation corpus and indexes, the TDT3 data and TDT2001 dry run indexes were used to compare relative system performance under various conditions. All runs performed used English training documents, reference document boundaries, English and translated Mandarin test documents, and manual transcripts of audio sources. The following control files from the dry run indexes were used:

```
trk_SR=nwt+bnman_TR=eng_TE=mul,eng_Nt=1.ctl
trk_SR=nwt+bnman_TR=eng_TE=mul,eng_Nt=2.ctl
trk_SR=nwt+bnman_TR=eng_TE=mul,eng_Nt=4.ctl
```

Several experiments were performed using $Nt = \{1|2|4\}$ and $Nn = \{0|2\}$. It was found that $Nt = 4, Nn = 0$ gives the best results, but possibly only because the lack of negative training examples causes the final threshold to be lower. In order to discriminate against irrelevant terms appearing in the positive examples, a parameter value of $Nn = 2$ was used for four of the five runs we submitted for the final evaluation.

3.1. Experiment Conditions

Runs were submitted for evaluation using the parameters shown in Table 1.

Run	Nt	Nn	TWB	Adap. Prof. Size	PRF Docs
a5042	4	2	4	50	–
p42	4	2	4	–	10
st42	4	2	4	–	–
th842	4	2	8	–	–
p10	1	0	10	–	10

Table 1: Run parameters for RMIT runs submitted.

4. Results

A comparison of results of the runs shown in Table 1 is shown in Table 2 (these are taken from the official NIST TDT evaluation). Statistics displayed are topic weighted and macroaveraged.

Run	C_{DET}	P_{miss}	P_{fa}
a5042	0.5171	0.5141	0.0006
p42	0.3809	0.3697	0.0023
st42	0.5267	0.5252	0.0003
th842	0.3839	0.3766	0.0015
p10	0.4764	0.4575	0.0038

Table 2: RMIT TDT 2001 Topic Tracking Results

Table 2 shows that our false alarm rates are very low, while the miss rates are high, resulting in poor C_{DET} scores. We believe that this is due to two factors:

1. Term expansion is not performed. This may cause our system to miss many documents because we do not have synonyms and related words that occur in on-topic test set documents

2. Our thresholding algorithm is simple, and appears to set thresholds somewhat higher than the optimum value

In Table 3, it can be seen that by lowering our threshold, we could have achieved less misses and more false alarms, with an overall improvement in our performance.

Run	$min\,(C_{DET})$	$min\,(P_{miss})$	$min\,(P_{fa})$
a5042	0.2785	0.2160	0.0128
p42	0.2244	0.1767	0.0097
st42	0.1939	0.1435	0.0103
th842	0.1939	0.1435	0.0103
p10	0.3636	0.2850	0.0160

Table 3: RMIT TDT 2001 Minimum Topic Tracking Results

Our primary evaluation condition run is shown in Figure 1.

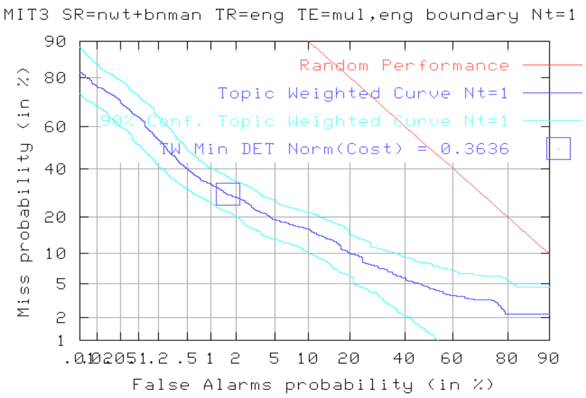

Figure 1: Top 10 pseudo-relevant documents added to profile (run "p10").

Our best evaluation run was `p42` and this is shown plotted in Figure 2.

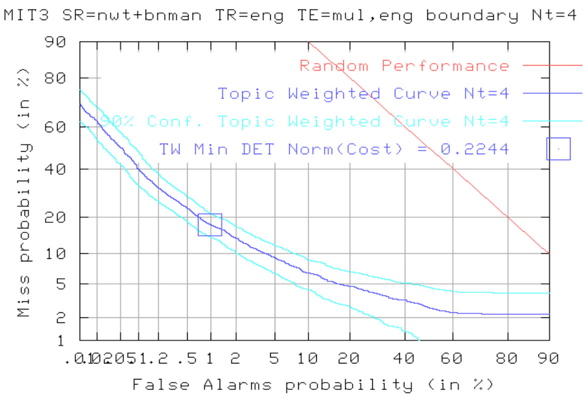

Figure 2: Top 10 pseudo-relevant documents added to profile (run "p42").

5. Discussion of Results

Our runs had high precision, as evidenced by the low false alarm rate. However, recall is the area of deficiency in our runs and this results in only moderate performance overall. One immediate improvement on recall, which can be observed by comparing runs `st42` and `th842`, is to lower the decision threshold.

Our best run used pseudo-relevance feedback. However, a closer examination of the minimum achievable C_{DET} shows that this is only because of the threshold point chosen. It appears that the `th842` and `st42` runs still have the best minimum C_{DET}. Normalising topic score distributions may prove an effective way to solve this problem, while a better source of pseudo-relevance feedback may be an older TDT corpus rather than TREC.

Adaptive profiling appears to have been partly successful, but it is apparent that there is still significant improvement possible. In particular, we have found that it is difficult to balance positive and negative feedback, and to keep the system stable. One obvious improvement would be addition of the ability to recalculate the threshold as the profile changes.

Static profiling has the advantage of being simpler to implement, and faster to run, as we skip all the steps of sorting and renormalising the active profile. However, static profiling is limited in that it cannot handle "concept drift" in a topic. We believe that it may be necessary to use adaptive profiling in order to improve our results further.

The adaptive profiling system used here implements a linear scaling function for profile adaptation, based on the certainty of a document being on or off topic. A better method may be to use a sigmoid function, so that borderline documents are totally ignored.

6. Conclusion and Future Directions

For this year's TDT evaluation, we developed a baseline topic tracking system. Some variations on the baseline were attempted, including adaptive tracking and pseudo-relevance feedback. While our system was efficient and based on our previous work in fast categorisation, the accuracy of the system can be improved substantially.

There are many possible future directions. The most promising directions for increasing accuracy are to improve the thresholding algorithm, normalise topic relevance scores, ambiguate topic profiles, apply higher weightings to proper names, and to recognise some multiword features. We also plan additional efficiency work.

7. Acknowledgements

This work was supported by the Australian Research Council, the Australian Defence Science and Technology Organisation, the Australian Defence Signals Directorate, and the Multimedia Database Systems group at RMIT University.

References

1. D. Harman. Overview of the second text retrieval conference (TREC-2). *Information Processing & Management*, 31(3):271–289, 1995.
2. T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *Proc. 14th International Conference on Machine Learning*, pages 143–151. Morgan Kaufmann, 1997.

3. D.D. Lewis, R.E. Schapire, J.P. Callan, and R. Papka. Training algorithms for linear text classifiers. In Hans-Peter Frei, Donna Harman, Peter Schäuble, and Ross Wilkinson, editors, *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 298–306, Zurich, Switzerland, 1996.
4. M.F. Porter. An algorithm for suffix stripping. *Program*, 13(3):130–137, 1980.
5. S.E. Robertson, S. Walker, S. Jones, M.M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In D.K. Harman, editor, *Proc. Text Retrieval Conference (TREC)*, pages 109–126, Gaithersburg, Maryland, 1994. NIST Special Publication 500-225.
6. J.J. Rocchio. Relevance feedback in information retrieval. In *The Smart Retrieval System — Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall, Englewood, Cliffs, New Jersey, 1971.
7. V. Shanks and H.E. Williams. Fast categorisation of large document collections. In *String Processing and Information Retrieval (SPIRE)*, San Rafael, Chile, 2001. To appear.
8. I.H. Witten, A. Moffat, and T.C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann Publishers, Los Altos, CA 94022, USA, second edition, 1999.