Broadcast Information Topic Segmentation
– BITS –

Yiu-Fai Cheung[a, b] Dietrich Klakow[b] Georg Bauer[b]
Dr. Drs. L.J.M. Rothkrantz[a]

[a] Delft University of Technology, Mekelweg 4, 2628 BZ Delft, the Netherlands
L.J.M.Rothkrantz@cs.tudelft.nl

[b] Philips Research, Weisshausstrasse 2, 52066 Aachen, Germany
Y.F.Cheung@its.tudelft.nl, Dietrich.Klakow@Philips.com, Georg.Bauer@Philips.com

Abstract

Thanks to the continuing progress in the Automatic Speech Recognition of Broadcast News (BN) audio streams it's possible to apply Information Retrieval techniques to the transcribed text, full of recognition errors. A Topic Segmentation task is necessary for this kind of systems to perform well. In this paper a new adapted solution approach, the BITS approach, is described for segmenting continuous BN streams into homogeneous topic/story segments inside Philips' Spoken Broadcast News Retrieval demonstrator system. This approach is based on Marti Hearst's TextTiling approach [1]. Some changes and improvements are made to overcome the weaknesses of the TextTiling approach working in the spoken text or transcription domain. The implemented prototype is tested and the test results will be reported in this paper.

1. Introduction

In the past few years, the Man-Machine Interface (MI) group of Philips Research Laboratory in Aachen has been working on the automatic recognition of BN audio streams. With the increasing improvement in the transcription a first step is made in the beginning of the year 2001 in building a Spoken Broadcast News Retrieval demonstrator system based on the recognized text. Topics in BN are marked by prosodic features or non-speech events. We transform speech in text using ASR. We have to realize that the outcome has on average 30-40% recognition errors.

In figure 1 we display a general information flow of our system. Data capturing of BN streams takes place at the start. Pure concentrating on the audio data stream a collection of transcribed text data is generated after processing by Philips' Automatic Speech Recognition (ASR) module. This will be the input for the Topic Segmentation task. The output results of the Topic Segmenter are the possible topic boundary positions found. These results will be stored in the Broadcast News Retrieval Database

together with the captured BN streams for retrieval purposes by the Document Retrieval module. The Dialogue Control system part provides the users an interface to communicate with the system by speech.

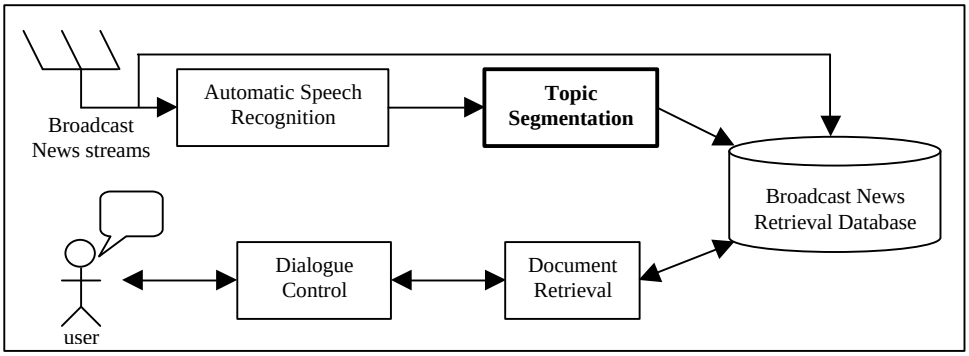

Figure 1. Philips' Spoken Broadcast News Retrieval demonstrator system architecture.

The focus of this paper is on the Topic Segmentation task. The main idea is to automatically divide a continuous BN (audio) stream into homogeneous topic/story segments. In other words, the positions inside the BN stream where the topic changes has to be found by such a Topic Segmenter. In general, this task is seen as a classification problem and can be described as a two-phase process: (1) presegment the continuous BN stream into very small (homogeneous) segments, i.e. all between segment positions are than candidate topic boundary positions, and (2) combine the small segments into larger homogeneous topic/story segments, i.e. classifying the candidate positions as topic and non-topic boundary positions.

2. Related Work

Classical Topic Segmentation approaches are focussed on pure lexical (i.e. text-based) information, such as TextTiling [1]. These solution approaches are only applied in the correct written text domain. In the BN data domain we'll be concentrating on the errorful spoken/recognized text domain. In that case, also the audio (i.e. prosodic-based) and TV/video (i.e. image-based) sources get involved. There will be problems when pure text-based approaches are applied in the BN data domain due to the lack of typographical cues (such as sentence ending markers and letter capitalizations).

Some other Topic Segmentation approaches make use of more than one group of topic boundary feature cues. Eichmann et al. [2] make use of text-based feature cues in the same way as Hearst's TextTiling [1]. Besides that this approach includes only the pause duration as prosodic feature cue to further improve the segmentation performance. Another approach is based on the idea of Shriberg et al. [3], where they combine text-based and a large set of prosodic feature cues together in three different combination methods. In the statistical approach of Beeferman et al. [7] it's even mentioned that all three groups of feature cues (text, prosody, and image) can be used in the same Topic Segmentation approach.

3. The Topic Segmenter

The new adapted TextTiling algorithm, i.e. the BITS approach, can be described in 8 steps. Each step of the algorithm will be described in more details below.

1. Find the candidate positions
2. Create the TextTile blocks
3. Perform preprocessing
4. Calculate the lexical cohesion
5. Enhancing the scores
6. Smoothing the output results
7. Perform depth scoring
8. Do threshold selection

3.1 Find the candidate positions

The first step is to narrow the space for doing the topic boundary calculations. It doesn't make any sense to calculate a topic boundary score at every between words position. The topic boundary calculations should only be done for the candidate topic boundary positions, i.e. the sentence boundary locations. These are indicated by the non-speech pause events in the generated transcriptions. Pause durations of longer than 0.3 seconds provide a good indication as candidate topic boundary positions.

3.2 Create the TextTile blocks

After locating the candidate topic boundary positions TextTile blocks are created. A TextTile block is just a collection of words. For each of the candidate positions a TextTile block on the left-hand side and on the right-hand side will be created. All TextTile blocks will have the same size (with the exception for the starting and ending data stream areas, where the amount of data will be less than the chosen TextTile block size). As a rule of thumb this block size is chosen equal to half the length of the average topic/story segment length in the BN domain (165 words), i.e. around 80 words.

3.3 Perform Preprocessing

Each TextTile block consists of the same amount of words, but not all words are of importance for doing the topic boundary calculation. This calculation is only interested in the content words (or keywords) that gives an impression of what the topic/story segment is about. Non-content words are usually commonly used terms, such as '*the*', '*on*' and '*a(n)*'. These terms could show up everywhere through the whole BN show, and not only at/around the topic/story segments that the system is interested in. Instead of using a stop word list to filter out the common terms, an Alembic tagger in combination with a lemmatizer tool developed by Philips' MI group is used to extract the keyword terms. The main reason for this change is that a domain specific stop word list is usually not available and most of the time it's not exhaustive enough.

3.4 Calculate the lexical cohesion

The topic boundary calculation mentioned previously, is the idea of calculating a similarity score between the left and right TextTile block on each candidate topic boundary position. The equation used for this calculation is given below. After doing the similarity scoring at all candidate positions, the results could be plotted in a so-called similarity graph or cohesion curve (figure 2). The lower the similarity value on this curve, the higher the probability that it's a topic boundary position.

The similarity score is calculated by the following equation:

$$\textit{Similarity score}(ltb, rtb, t) = \frac{\Sigma_t (w_{t,ltb} * w_{t,rtb})}{\sqrt{\{ \Sigma_t (w_{t,ltb})^2 * \Sigma_t (w_{t,rtb})^2 \}}}$$

"ltb" : left TextTile block
"rtb" : right TextTile block
"t" : candidate topic boundary position

Where t ranges over all keyword terms in the two TextTile blocks, and $w_{t,[x]}$ is the weight assigned to each term in the TextTile block [x]. In this version of the similarity score calculation, the weights on the terms are simply its frequency of occurrence within its TextTile block. This equation will yield a similarity score between 0 and 1 after normalization by the denominator term.

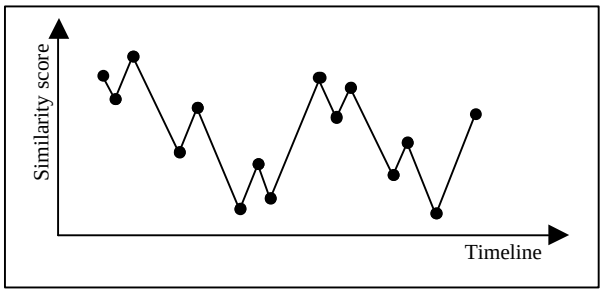

Figure 2. An illustrative example of a similarity graph or cohesion curve.

3.5 Enhancing the scores

The resulting scores from the previous step can now be used for finding the desired topic boundary positions. But there are still some topic boundary positions missing or falsely identified. The cohesion curve is not always as strong as it looks like or should be. One main reason is that the original TextTiling solution was meant for the correct written text domain. Some improvement can be made after this step. The idea of the original TextTiling approach is that all information needed should be extractable from the cohesion curve. The improvement approaches are thus based on enhancing the similarity scores found in the previous step to change the cohesion curve into the desired form.

There are three types of improvements that can be done. For this prototype Topic Segmenter only the first two improvements listed below are implemented.

1. **Topic Pause Improvement**: each candidate position is already a sentence pause position. Long pauses have a high probability to indicate topic boundary positions in the BN domain. A similarity score is known for each candidate position. Based on the topic pause length, this score can be enhanced by scaling it down. Remember the fact that the lower the similarity score, the higher the probability to be a topic boundary location. This could help to select the desired topic boundary position, because the original TextTiling approach will usually provide a small area with more than one candidate position. Pause events that are very long (e.g. 4.0 seconds) are already used for making hard decisions to be a topic boundary position [2].
2. **Cue Word Phrase Improvement**: some repetitive structures of word phrases shows up at the beginning and at the end of a topic/story segment in the BN domain [3]. These lexical discourse or cue word phrases can be detected by looking at matches of word strings at the left and at the right hand side of each candidate position within a small block of 20 words (i.e. the average BN sentence length). The similarity score at that position can be scaled down again, when such a cue word phrase is detected. The scaling effect could be made stronger if cue word phrases are detected on both side of the candidate position. This could also help to locate the desired topic boundary position, when the exact location is not clear.
3. **Semantic Improvement**: the original approach was based on correct written text, and the amount of keywords in each TextTile block is limited. There are even fewer words that can be used in this task domain of errorful spoken/recognized text. The cohesion curve will be weaker, when used in this text domain. Two approaches are known in the literature to overcome this problem, i.e. Local Context Analysis (LCA) [4] and Latent Semantic Analysis (LSA) [5]. The idea is to substitute the keyword terms in each TextTile block by semantically related terms to enlarge the amount of data being used. With this approach even the smallest TextTile segment of one sentence long can be compared to each other in the TextTiling approach. With the addition of this improvement in the BITS approach it should also be possible to find detailed information inside (large) topic/story segments. We will implement this in the next future. Adding this improvement won't change the algorithm. It will only have (a better) effect on the similarity scores, and provide us the possibility to use smaller TextTiles.

3.6 Smoothing the output results

In general, there are much more sentence boundary positions (or pause events) than (sub)topic boundary positions. The look of the cohesion curve is usually very noisy because of the many positions where similarity calculation has taken place. It's than not clear from the cohesion curve which position to choose as topic boundary position. Some simple average filtering could smoothen this cohesion curve to make the final step in the BITS approach, i.e. selecting the desired topic boundary positions, easier to perform. After average smoothing filtering it should be clear from the cohesion curve, around which area a topic boundary is likely to be found. A weak point of this filtering process is of course that it lacks the ability to find the topic boundary position at the desired position. Only simple average smoothing filters of small sizes (e.g. 3 or 5) are useful.

3.7 Perform depth scoring

Unfortunately, no topic boundary decisions could be made based on the absolute similarity scores of the cohesion curve. The main reason for this is because the similarity calculation is strongly dependent and varies within the BN stream. So some areas could give all high similarity scores and other areas all very low similarity scores on the curve. But it still provides a clear picture of the places where to find the topic boundaries. It's the change in the cohesion curve that really matters. The stronger the change in similarity scores, the higher the probability to be a topic boundary position. This effect is measured by depth scoring. The depth score could be calculated as mentioned below for each valley/gap position in the cohesion curve. Furthermore, it seems that the previous average smoothing filtering step is very important for the depth scoring step. Without that a much lower and unwanted depth score will usually be the result. By some simple average smoothing filtering, the local unwanted effect due to the many candidate positions is filtered out (see figure 3).

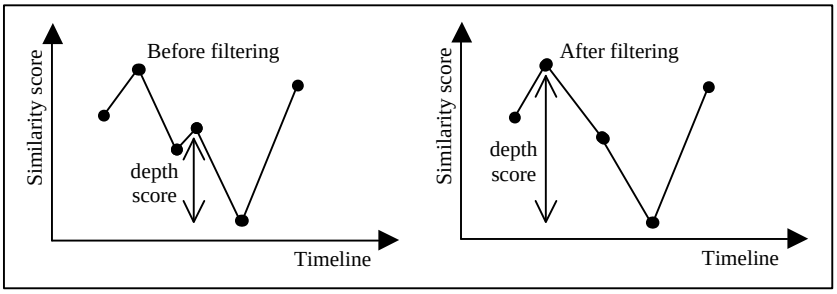

Figure 3. An example illustrating the depth score calculation before and after filtering.

3.8 Do threshold selection

Not all positions where a depth score is calculated are topic boundary positions. Only the positions which pass some threshold have a high probability to be a real topic boundary position. This final step of the BITS approach is called the threshold selection step. So some threshold value has to be determined. This value could be chosen automatically. The threshold value could be made as a function of the depth score characteristics for a given data domain, by using the average, $\langle s \rangle$, and the standard deviation, σ, of their scores. An assumption is made here that the scores are normally distributed. Depending on the situation a threshold depth score value of $\langle s \rangle - \sigma$ or $\langle s \rangle - \sigma/2$ can be applied.

4. Segmentation Evaluation

The performance of the BITS approach is tested on a corpus of 8,5 hours of 17 CNN BN shows. The (sub)topics for these CNN news shows are manually found. The standard TDT based performance metric for the BN domain is used for doing quantitative segmentation result evaluations [6]. This measure takes into account of the closeness of incorrectly identified topic boundaries to actual topic boundaries, and does not treat all incorrect boundaries the same. The input for this metric is the manually segmented

results and the automatically generated results by the BITS approach. The miss error ratio (P_{miss}) and the false alarm error ratio (P_{Fa}) will be calculated first. The total topic segmentation error ratio (C_{seg}) could be obtained by the following standard equation:

$$C_{seg} = C_{miss} \times 0.3 \times P_{miss} + C_{Fa} \times 0.7 \times P_{Fa}$$

It's still not clear how to choose the C_{miss} and C_{Fa} values. In this application domain it seems that misses are much worse than false alarms, because a miss will provide us a large inhomogeneous topic/story segment (not wanted by retrieval systems) and on the other hand a false alarm will still provide us (smaller) homogeneous topic/story segments. Intuitively, the C_{miss} factor should be chosen higher than the C_{Fa} factor. This field is still open to be investigated. The C_{seg} results in this section are calculated by choosing $C_{miss} = C_{Fa} = 1$. To check the changes when $C_{miss} > C_{Fa}$ is chosen, also the total of the two error types are added together ($P_{miss} + P_{Fa}$) for the situations before and after improvements is applied in the BITS approach (see below).

Table 1. Results of test experiments with topic pause improvement only.

	No improvement	After topic pause improvement
Average P_{miss}	51.74%	27.29%
Average P_{Fa}	33.58%	48.26%
Average C_{seg}	39.02%	41.96%
Average $P_{miss} + P_{Fa}$	85.32%	75.55%

Table 2. Results of test experiments with cue word phrase improvement only.

	No improvement	After cue word phrase improvement
Average P_{miss}	51.74%	48.30%
Average P_{Fa}	33.58%	33.79%
Average C_{seg}	39.02%	38.14%
Average $P_{miss} + P_{Fa}$	85.32%	82.09%

Table 3. Results of test experiments combining topic pause and cue word phrase improvements.

	No improvement	After topic pause + cue word phrase improvements
Average P_{miss}	51.74%	27.68%
Average P_{Fa}	33.58%	46.60%
Average C_{seg}	39.02%	40.92%
Average $P_{miss} + P_{Fa}$	85.32%	74.28%

It's clear from table 1, 2 and 3, that if one error type decrease the other will increase. A trade off has to be made between the two error types. Furthermore, the C_{seg} result with $C_{miss} = C_{Fa} = 1$ doesn't provide us a significant difference between the situations with and without improvements. But by looking at the combined error ratios (i.e. $C_{miss} > C_{Fa}$) it provide us the best segmentation results (i.e. lowest error ratios).

5. Conclusions

A new adapted solution approach is applied into Philips' Spoken Broadcast News Retrieval demonstrator system based on the well-known TextTiling approach. A first prototype including improvements has been implemented. By comparing the segmentation results before and after adding the improvement approaches mentioned, the idea of the BITS approach seems to work as expected. Especially the situation of combining different improvement approaches will further increase the segmentation performance. In the next Topic Segmenter version, the semantic improvement approach should be implemented to find subtopic or detailed information inside the BN stream. Even future feature cues, such as TV/video-based cues, can be included in the same way in the BITS approach to further improve the segmentation performance.

References

[1] M. A. Hearst. *TextTiling: Segmenting Text into Multi-Paragraph Subtopic Passages*. 1997.

[2] D. Eichmann, M. Ruiz, P. Srinivasan, N. Street, C. Culy, and F. Menczer. *A Cluster-Based Approach to Tracking, Detection and Segmentation of Broadcast News*, in Proceedings of the DARPA Broadcast News Workshop February 28 – March 03, 1999.

[3] E. Shriberg, A. Stolcke, D. Hakkani-Tür, and G. Tür. *Prosody-Based Automatic Segmentation of Speech into Sentences and Topics*, Speech Communication 32(1-2), pages 127-154 (Special Issue on Accessing Information in Spoken Audio), 2000.

[4] J. M. Ponte, W. B. Croft. *Text Segmentation by Topic*, in Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries, pages 120-129, 1997.

[5] F. Y. Y. Choi, P. Wiemer-Hastings, and J. Moore. *Latent Semantic Analysis for Text Segmentation*, 2001.

[6] *The Topic Detection and Tracking Phase 2 (TDT2) Evaluation Plan*, 1998.

[7] D. Beeferman, A. Berger, and J. Lafferty. *Statistical Models for Text Segmentation*, 1999.