Segmenting Documents Using Multiple Lexical Features

Amanda C. Jobbins and Lindsay J. Evett
Department of Computing
Nottingham Trent University
Nottingham NG1 4BU, UK
ajobbins@resumix.com and lje@doc.ntu.ac.uk

Abstract

A method is presented for segmenting documents into conceptually related areas. Determining the equivalence of text is often based on the number of word repetitions. This approach is unsuitable for detecting short segments because terms tend not to be repeated across just a few sentences. In this paper we investigate the contribution of two other lexical features to find related words: collocation and relation weights (which identify semantic relations). An experiment was conducted on a set of test data with known topic changes; performances of the three features were independently compared. A combination of all features was the most reliable indicator of a topic change. In another experiment, CNN news summaries were segmented into their individual news stories. Precision and recall rates of around 90% are reported for news story boundary detection.

1. Introduction

In this paper we investigate the conceptual segmentation of a document. This method is useful for a number of applications such as text summarisation and information retrieval. A document can be segmented, each segment independently summarised, then all summaries combined to produce an abstract. To retrieve just the relevant areas of a document, text segmentation can decompose a document into related sections and then query terms can be compared to each section.

Typically, segmentation algorithms identify related text segments by matching on repeated words [1, 2]. This method is unreliable for segmenting between short excerpts such as a few sentences [3, 4]. Ponte and Croft [4] segmented a database of brief news broadcasts. To increase the number of terms for matching, existing terms were expanded with collocations. (A collocation is a sequence of words, usually pairs, that frequently co-occur.) Another segmenter actually matched on collocating words [5]. Semantic relations derived from WordNet have been used for text segmentation [6, 7] and for document matching [8]. Many of these approaches failed to improve on the performance of algorithms that rely on word repetition.

It was proposed that the reliability of segments found could be improved, particularly for short segments, by looking for multiple features. Lexical cohesion [9] describes the semantic relations that exist between words in a text. Sections of text that are strongly cohesive (have many relations) are likely to be related in meaning. This behaviour is useful for text segmentation. Lexical cohesion is generally represented in segmentation algorithms by just word repetition. Word repetition does comprise a significant proportion of lexical cohesion—nearly three-quarters of all ties [9]—but it is not the only contributing factor. Collocation accounts for about 17% of lexical cohesion ties, and a further 10% are synonym or superordinate relations. Morris and Hirst's [10] segmenter looked for word repetition and semantic relations derived from a thesaurus. However, it did not include collocation, and was not automated. This paper presents a text segmentation algorithm that employs multiple lexical features, so short segments can be detected.

2. Proposed algorithm

In earlier work [11] a text segmentation algorithm was described that captured all types of lexical cohesion ties. To automatically find ties between pairwise words three features were developed: word repetition, collocation and relation weights. This paper describes the use of a modified segmenter and its application to short segments. Modifications include the automatic detection of sentences; the normalisation of all scores, so each feature contributes equally; and the summation of association ratios for collocations, rather than cumulating the number of

occurrences. These enhancements have improved the segmenter's performance.

The proposed algorithm for text segmentation is shown in Figure 1. The algorithm uses 'blocking.' Lexical similarity is calculated for adjacent blocks of sentences, and segment boundaries are placed between blocks with low similarity. Currently, block size is variable which is useful for dealing with different length segments.

1. Locate sentence boundaries.
2. Compare pairwise words across adjacent blocks:
   2.1 Ignore function words.
   2.2. Find related words (ties) using lexical features:
   Word repetition;
   Collocation;
   Relation weights.
   2.3. Calculate a feature score for each matching pair.
3. Calculate similarity scores:
   3.1. Cumulate feature scores for each lexical feature across adjacent blocks.
   3.2. Normalise feature scores across all blocks.
   3.3. If multiple features are used, combine feature scores across adjacent blocks then re-normalise.
4. Insert segment boundaries at troughs in the similarity scores.

Figure 1. Segmentation algorithm.

Word repetition ties are identified by identical word pairs and pairs with the same root such as *dark* and *darker*. Morphological analysis is done by consulting a lexicon of root and inflected pairs (e.g. *dark darker*). A word pair is identified as a collocation by locating it in a lexicon comprising collocations and their association ratios [12] such as $I(dark,ages) = 7.47$. Relation weights [13] identify and weight (0 to 100) semantically related pairs. They are based on the lexical organisation of Roget's Thesaurus where both superordinate and synonym relationships are represented. About 20% of all word pairs (x,y) compared attain a significant weight where $RW(x,y) > 0$. However, only strongly related pairs attain weights where $RW(x,y)$ approaches 100, for example, $RW(church,priest) = 81.25$.

A *feature score* is calculated for each matching word pair, $f(x,y)$. Word repetition scores are quantitative; the number of repetitions observed are cumulated. Collocation and relation weight scores are qualitative; both features measure the strength of association between words. Feature scores are cumulated for each lexical feature across adjacent blocks by $\frac{1}{N}\Sigma f(x,y)$ where N is the total number of words compared. A similarity score is calculated for each pair of adjacent blocks based on the feature scores for that pair. A trough in a sequence of similarity scores across a text signals a potential change of topic (a shift in the subject area discussed). The current algorithm considers all troughs to be an indication of a topic change. In future work a threshold or filter could be applied to differentiate between troughs. Hearst [1] selected troughs according to their relative depths.

3. Experimental results

Two experiments are reported here which investigate the performance of the segmentation algorithm. In the first experiment a set of test data was generated to represent the base case. Pairwise articles from different topics were concatenated, so each concatenated text had at least one (known) topic change. The assumption was made that the location of these engineered changes was the easiest case. In the second experiment, CNN news summaries were segmented; this data is a real example of compounded text.

3.1. Locating known topic changes

Ten topical articles, each covering a distinct subject, were collected from the Web. Concatenating pairs of these articles generated a total of 90 texts for test data. The transition from the first article to the second article represented a *known topic change*. The segmentation algorithm was applied using the three features both individually and in combination, and with a block size of six sentences following Hearst [1]. Table 1 gives the results for the comparison of troughs placed by the segmentation algorithm to the known topic changes in the texts.

feature set used	mean number troughs per text	no error margin		one sentence error margin	
		changes found	prob.	changes found	prob.
coll, rep, RW	4.0	86	.18	90	.53
rep, RW	3.6	84	.16	90	.47
coll, rep	4.0	83	.17	89	.52
rep	3.5	82	.15	90	.45
coll, RW	4.5	71	.20	85	.59
RW	5.1	65	.22	78	.67
coll	4.3	58	.19	81	.56

Table 1. Known topic changes found in 90 generated texts using a block size of six.

The table shows the mean number of troughs found per text; the number of troughs that coincide with a known topic change; and the probability of a trough and a known change coinciding. Probability was calculated by dividing the number of troughs placed in a text by the total number of troughs that could occur.

The majority of known topic changes coincided with a trough (96%) when all features were looked for across a text. This result is impressive because no error margin was used, and it reduces by approximately 50% the error rate of the best single feature—word repetition. Only four known topic changes went undetected. For each of these error cases a trough was placed within one sentence of the topic change. Hence, with a one sentence error margin, all 90 known topic changes were located. However, the probability of finding a change increases significantly when allowing an error margin.

Looking for multiple features outperforms the use of these features individually. Word repetition was the most successful feature when applied alone (91%). This result is not surprising. Much previous work has used word repetition, so evidently it is a significant indicator of related text. This experiment demonstrates that word repetition in conjunction with additional lexical features gives a better boundary detection rate.

3.2. Segmenting CNN news summaries

The objective of the current investigation was to determine if all troughs in a text coincide with topic changes, and to test the algorithm with real-world data. Sixty CNN news summaries were collected at random from the Web (http://cnn.com/QUICKNEWS/print.html) for test data. The data consisted of 2,019 news stories (segments) giving 1,959 segment boundaries for detection. These boundaries were considered the ground truth—the only topic changes in the test data.

In the previous experiment, a block size of six sentences was used. For the current test data, with an average segment size of 3.3 sentences, this block size is too large. So, the algorithm was tested with block sizes ranging from one to four sentences. The combination of all features produced the best correct rate in the first experiment; therefore, the same configuration was adopted in this experiment.

The troughs placed by the algorithm were compared to the news story boundaries. Table 2 shows the statistical mean results for the segmentation of the summaries. Precision and recall are given for both the exclusion of move errors (not allowing an error margin) and the inclusion of move errors when an error margin of one sentence was considered.

		block size			
		1	2	3	4
P(trough at boundary)		.32	.29	.28	.27
troughs per text		35.1	31.0	29.1	28.0
boundaries found		21.2	21.8	20.3	19.0
insertion errors		6.0	2.7	2.9	3.0
deletion errors		3.5	4.4	6.5	7.7
move errors		7.9	6.4	5.9	5.9
precision	(excluding move errors)	60.4%	70.4%	69.5%	68.2%
recall	(excluding move errors)	64.9%	66.8%	61.9%	58.2%
precision	(including move errors)	83.1%	91.3%	90.4%	89.6%
recall	(including move errors)	89.1%	86.5%	80.1%	76.4%

Table 2. CNN news summaries segmented using different block sizes.

With a block size of two, 66.8% of the boundaries were detected. On average, 6.4 boundaries per summary were within one sentence of a trough. Including these move errors reduces the error rate by nearly 60% giving a 86.5% boundary detection rate. In Figure 2, which shows the segmentation results for one of the CNN news summaries used for test data, there are four move errors. All other troughs (27) in this summary coincided exactly with a boundary.

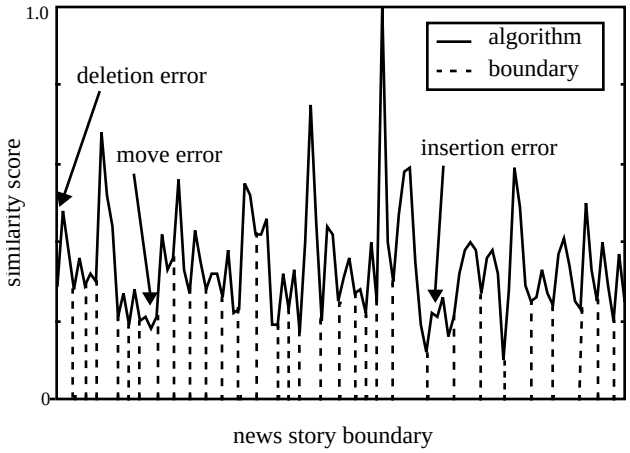

Figure 2. Position of news story boundaries in a CNN news summary in relation to troughs found by the algorithm.

Four summaries had 100% precision where all troughs corresponded to a news story boundary. However, none of the summaries attained perfect recall where all boundaries were found. Some news stories were only two sentences long. With a block size of the same length or longer, these stories were always compared alongside sentences from adjacent, unrelated stories. Consequently, the segment boundaries were more difficult to distinguish. Also, news stories are grouped together by their relevance to a particular category such as world news and politics. In some cases boundaries went undetected because consecutive news stories had similar subject matter.

A block size of one has the most insertion errors (6.0). An example of this error type is given in Figure 2. The trough at sentence 72 does not coincide with a boundary. This trough, however, is weak (has a shallow valley) and could be eliminated by thresholding. The larger block sizes (three and four) tend to under-segment, so more deletion errors occur. Boundaries were missed at the start of a summary because block size was too coarse. In Figure 2, the initial boundary at sentence three went undetected because it was in the first block comparison (i.e. sentences one and two compared to sentences three and four).

The current algorithm, which incorporates semantic information, improves on systems that look for just word repetition or just collocation (e.g. [1, 5]). Ponte and Croft [4] used collocations to expand terms. This approach worked well, achieving precision and recall rates of 95.0% and 84.4% (with no error margin), increasing to 95.9% and 85.2% with a two sentence margin. They reported that term expansion processes 200 KB of text per hour. The current segmenter has a faster processing speed (over 700 KB per hour), so it is more suitable for systems where response time is relevant.

4. Conclusions and future work

The first experiment demonstrated that word matching using semantic relations, in addition to word repetition, improves segmentation. A total of 96% of the known topic changes in 90 texts were located. Four topic changes were not found, but in each case a trough was placed within one sentence of the change. In the second experiment, short segments were successfully detected in CNN news summaries. Nearly 70% of the boundaries between news stories were detected. Including move errors (boundaries within one sentence of a trough) improved boundary detection to 86.5%.

The segmentation algorithm worked well for different segment lengths. In the first experiment segments averaged 17 sentences, and in the second experiment they were 3.3 sentences. The only parameter changed between the two experiments was block size. Currently, block size has been chosen by examining the data. In future work, an automatic process could be run where several block sizes are tested and a characteristic of the result could be analysed to determine the most suitable block size to proceed.

An issue not addressed in this research is the consideration of document formatting features. Web-based CNN news summaries, for instance, include the section headers “World News” and “U.S. News.” These headers indicate the start of a new section and hence the beginning of a new story; they could be used to cue segment boundaries. However, such cues are likely to be document specific. A set of boundary cues would have to be identified for every document type processed by the segmenter.

References

1. M.A. Hearst (1994) Multi-paragraph segmentation of expository texts, *Report No. UCB/CSD 94/790*, University of California, Berkeley

2. Y. Yaari (1997) Segmentation of expository texts by hierarchical agglomerative clustering, *RANLP’97*, Bulgaria

3. G. Salton and C. Buckley (1991) Global text matching for information retrieval, *Science*, 253, pp. 1012-1015

4. J.M. Ponte and W.B. Croft (1997) Text segmentation by topic, *Proceedings of the 1st European Conference on Research and Advanced Technology for Digital Libraries*, pp. 113-125

5. D. Beeferman, A. Berger and J. Lafferty (1997) Text segmentation using exponential models, *Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing*

6. M.A. Hearst (1993) TextTiling: A quantitative approach to discourse segmentation, *Technical Report 93/24*, Sequoia 2000, University of California, Berkeley

7. M.A. Stairmand (1997) Textual context analysis for information retrieval, *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, Philadelphia, pp. 140-147

8. S.J. Green (1998) Automated link generation: Can we do better than term repetition?, *Proceedings of the Seventh International World Wide Web Conference*, Brisbane, Australia, pp. 75-84

9. M.A.K. Halliday and R. Hasan (1976) *Cohesion in English*, Longman Group

10. J. Morris and G. Hirst (1991) Lexical cohesion computed by thesaural relations as an indicator of the structure of text, *Computational Linguistics*, 17(1), pp. 21-48

11. A.C. Jobbins and L.J. Evett (1998) Text segmentation using reiteration and collocation, *17th International Conference on Computational Linguistics*, Montreal, Canada

12. K.W. Church and P. Hanks (1990) Word association norms, mutual information and lexicography, *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pp. 76-83

13. A.C. Jobbins and L.J. Evett (1995) Automatic identification of cohesion in texts: Exploiting the lexical organisation of Roget’s Thesaurus, *Proceedings of ROCLING VIII*, Taipei, Taiwan