Detecting Subject Boundaries Within Text: A Language Independent Statistical Approach

Korin Richmond (korin@cogsci.ed.ac.uk)
Andrew Smith (ajs@cogsci.ed.ac.uk)
Einat Amitay (einat@cogsci.ed.ac.uk)

(Joint Authorship)
Centre for Cognitive Science
2 Buccleuch Place
Edinburgh EH8 9LW
SCOTLAND

Abstract

We describe here an algorithm for detecting subject boundaries within text based on a statistical lexical similarity measure. Hearst has already tackled this problem with good results (Hearst, 1994). One of her main assumptions is that a change in subject is accompanied by a change in vocabulary. Using this assumption, but by introducing a new measure of word significance, we have been able to build a robust and reliable algorithm which exhibits improved accuracy without sacrificing language independency.

1 Introduction

Automatic detection of subject divisions within a text is considered to be a very difficult task even for humans, let alone machines. But such subject divisions are used in more complex tasks in text processing such as text summarisation. An automatic method for marking subject boundaries is highly desirable. Hearst (Hearst, 1994) addresses this problem by applying a statistical method for detecting subjects within text.

Hearst describes an algorithm for what she calls *TextTiling*, which is a method for detecting subject boundaries within a text. The underlying assumption of this algorithm is that there is a high probability that words which are related to a certain subject will be repeated whenever that subject is mentioned. Another basic assumption is that when a new subject emerges the choice of vocabulary will change, and will stay consistent within the subject boundaries until the next change in subject. These basic notions of vocabulary consistency within subject boundaries lead to a method for dividing text based on calculating vocabulary similarity between two adjacent windows of text.

Each potential subject boundary is identified and assigned a correspondence value based on the lexical similarity between two windows of text, one on either side of the subject boundary. The values for all potential boundaries are plotted on a graph, creating peaks and troughs. The troughs represent changes in vocabulary use and therefore, according to the underlying assumption, a change in subject. A division mark is inserted where a significant local minimum is detected on the graph. Hearst measured approximately 80% success in detection of subject boundaries on some texts.

We decided to adopt Hearst's underlying assumption that a change in subject will entail a change in vocabulary. Our aim was to make the algorithm as language independent and computationally expedient as possible, while also improving accuracy and reliability.

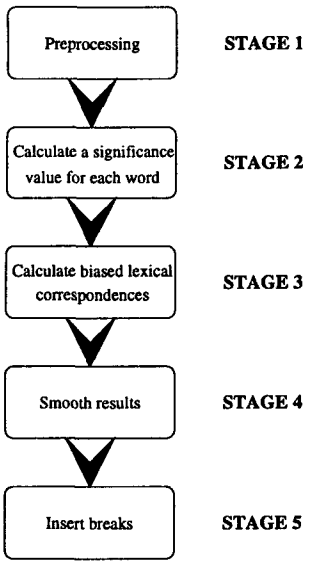

Figure 1: Algorithm Structure.

2 Design

The algorithm is divided into five distinct stages. Figure 1 shows the sequential, modular structure of the algorithm. Each stage of the algorithm is described in more detail below.

2.1 Preprocessing (stage 1)

In her implementation of the *TextTiling* algorithm Hearst ignores preprocessing, claiming it does not affect the results (Hearst, 1994). By preprocessing we mean lemmatizing, stemming, converting upper to lower case etc. Testing this assumption on her algorithm indeed seems not to change the results. However, using preprocessing in conjunction with stage 2 of our algorithm, does improve results. It is important for our algorithm that morphological differences between semantically related words are resolved, so that words like "bankrupt" and "bankruptcy", for example, are identified as the same word.

2.2 Calculating a significance value for each word (stage 2)

Hearst treats a text more or less as *a bag of words* in its statistical analysis. But natural language is no doubt more structured than this. Different words have differing semantic functions and relationships with respect to the topic of discourse. We can broadly distinguish two extreme categories of words; *content* words versus *function* words. Content words introduce concepts, and are the means for the expression of ideas and facts, for example nouns, proper nouns, adjectives and so on. Function words (for example determiners, auxiliary verbs etc.) support and coordinate the combination of content words into meaningful sentences. Obviously, both are needed to form meaningful sentences, but, intuitively, it is the content words that carry most weight in defining the actual topic of discourse. Based on this intuition, we believe it would be advantageous to identify these content words in a text. It would then be possible to bias the calculation of lexical correspondences (stage 3) taking into account the higher significance of these words relative to function words.

We would ideally like firstly to reduce the effect of noisy non-content words on the algorithm's performance, and secondly to pay more attention to words with a high semantic content. In her implementation, Hearst attempts to do this by having a finite list of problematic words that are filtered out from the text before the statistical analysis takes place (Hearst, 1994). These problematic words are primarily function words and low semantic content words, such as determiners, conjunctions, prepositions and very common nouns.

Church and Gale (Church and Gale, 1995) mention the correlation between a word's semantic content and various measures of its distribution throughout corpora. They show that: *"Word rates vary from genre to genre, topic to topic, author to author, document to document, section to section, paragraph to paragraph. These factors tend to decrease the entropy and increase the other test variables"*. One of these *other test variables* mentioned by Church and Gale is *burstiness*. They attribute the innovation of the notion of burstiness to Slava Katz, who, pertaining to this topic, writes (Katz, 1996): *"The notion of burstiness... will be used for the characterisation of two closely related but distinct phenomena: (a) document-level burstiness, i.e. multiple occurrence of a content word or phrase in a single text document, which is contrasted with the fact that most other documents contain no other instances of this word or phrase at all; and (b) within-document burstiness (or burstiness proper), i.e. close proximity of all or some individual instances of a content word or phrase within a document exhibiting multiple occurrence."* Katz has highlighted many interesting features of the distribution of content words, which do not conform to the predictions of statistical models such as the *Poisson*. Katz (Katz, 1996) states that, when a concept named by a content word is topical for the document, then that content word tends to be characterised by multiple and bursty occurrence. He claims that, while a single occurrence of a topically used content

word or phrase is possible, it is more likely that a newly introduced topical entity, will be repeated, *"if not for breaking the monotonous effect of pronoun use, then for emphasis or clarity"*. He also claims that, unlike function words, the number of instances of a specific content word is not directly associated with the document length, but is rather a function of how much the document is about the concept expressed by that word.

Therefore, the characteristic distribution pattern of topical content words, which contrasts markedly with that of non-topical and non-content words, could provide a useful aid in identifying the semantically relevant words within a text. Brief mention should be made of the work done by Justeson and Katz (Justeson and Katz, 1995), which, to a certain degree, relates to the requirements of our task. In their paper, Justeson and Katz describe some linguistic properties of technical terminology, and use them to formulate an algorithm to identify the technical terms in a given document. However, their algorithm deals with complex noun phrases only, and, although the technical terms identified by their algorithm are generally highly topical, the algorithm does not provide the context sensitive information of how topical each incidence of a given meaningful phrase is, relative to its direct environment. It is precisely this information that is needed to judge the content of a particular segment of text.

Although Katz (Katz, 1996) acknowledges what he calls two distinct, but closely related, forms of burstiness, he concentrates on modelling the inter-document distributions of content words and phrases. He then uses the inter-document distributions to make inferences about probabilities of the repeat occurrences of content words and phrases within a single document. Another divergence between what Katz has done so far and what the task of subject boundary insertion requires, is that he decides to ignore the issues of coincidental repetitions of non-topically used content words and simply equates *"single occurrence with non-topical occurrence, and multiple occurrence with topical occurrence."*

We have implemented a method which assigns an estimated significance score based on a measure of two context dependent properties; local burstiness and global frequency. The heart of our solution to the problem of assigning context-based values of topical significance to all words in a text, can be summed up in the following formula:

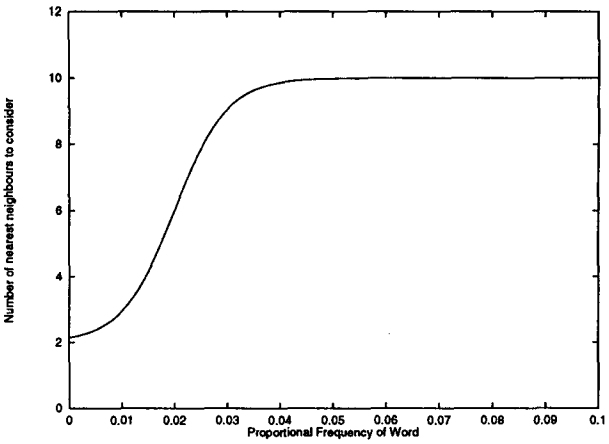

Figure 2: Calculation of number of nearest neighbours.

$$significance(x) = \frac{1}{n} \times \sum_{i=1}^{n} arctan\left(\frac{D_{x,i}}{W/\omega}\right) \quad (1)$$

where x is an individual word in the document and $D_{x,i}$ is the distance between word x and its ith nearest neighbour. The 1st nearest neighbour of word x is the nearest occurrence of the same word. The 2nd nearest neighbour of x is the nearest occurrence of the same word ignoring the 1st nearest neighbour. In general, the ith nearest neighbour of x is the nearest occurrence of the same word ignoring the 1st, 2nd, 3rd,...,$(i-1)$th nearest neighbours. W is the total number of words in the text. ω is the number of occurrences of the word like x. n is the number of nearest neighbours to include in the calculation and depends on the overall frequency of the word in the text. This formula will yield a significance score that lies within the range 0 to $\frac{\pi}{2}$ (high significance to low significance). This number is then normalised to between 0 and 1, with 0 indicating a very low significance, and 1 indicating a very high significance. The exact value of n is calculated separately for each distinct word, using the following formula:

$$n = \left(\frac{8}{1 + e^{-200\left(\frac{\omega}{W} - 0.02\right)}}\right) + 2 \quad (2)$$

This is essentially a sigmoid function with the range varying between two and ten, as shown in Figure 2. The constants scale and translate the function to yield the desired behaviour, which was derived empirically. The number of nearest neighbours to consider in equation 1 increases with the word's frequency. For example, when calculating the signif-

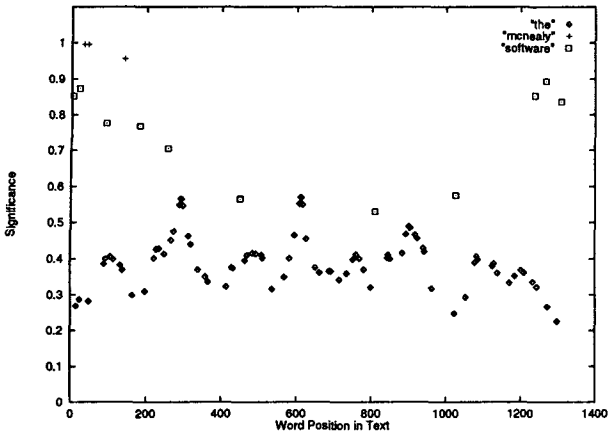

Figure 3: Significance Values.

icance of the least frequent words, only two nearest neighbours are considered. But for the most frequently occurring words, the number of nearest neighbours is ten. Figure 3 shows the main features of the performance of this significance assignment algorithm when tested on a sample text. The results for three very different words are shown.

Two general trends are the most important features of this graph. Firstly, elevated significance scores are associated with local clusters of a word. For example the cluster of three occurrences of "software" (a content word) at the end of the document have high significance scores. This contrasts with the relatively isolated occurrences of the word "software" in the middle of the document, which are deemed to be little more significant than several occurrences of the word "the" (a function word). Secondly, frequent words tend to receive lower significance scores. For example, even local clusters of the word "the" only receive relatively low significance scores, simply because the word has a high frequency throughout the document. Conversely, "McNealy" (a high semantic content word), which only occurs in a cluster of three, receives a high significance value. The important result shown by the graph is that content words (real names such as "McNealy") receive higher significance values than function words ("the").

We found that an *optimal* solution to the problem of balancing local density against global frequency was rather elusive. For example, the words at the centre of a cluster automatically receive a higher score, whereas it may be more desirable to have all the members of a cluster assigned a score lying in a narrower range. There are many other contentious issues which need to be investigated, such as the use of the ratio of all the occurrences of a word in a given text to the total length of that text in order to calculate the relative significance measure. Based on intuition, partly derived from Katz's discussion (Katz, 1996) of the relationship between document length and word frequency, the exact nature of this relationship across various document lengths may not be reliable enough. It may be more consistent to consider this ratio within a constant window size, e.g. 1000 words.

The advantage of this simple statistical method of distinguishing significant content words from non-content words is that no words need to be removed before allowing the algorithm to proceed. The output of this stage is a normalised significance score (0-1) for each word in the text. This significance score can then be taken into account when analysing the text for subject boundaries.

2.3 Calculate Biased Lexical Correspondences (stage 3)

Let us consider two sets of words, set A and set B. The main aim of this stage of the processing is concerned with calculating a correspondence measure between two such sets depending on how similar they are, where similarity is defined as a *measure of lexical correspondence*. If many words are shared by both set A and B, then the lexical correspondence between the two sets is high. If the two sets do not share many words, then the correspondence is low. Now let A' be the subset of A that contains only those words that occur somewhere in B. And let B' be the subset of B that contains only those words that occur somewhere in A. The lexical correspondence between sets A and B can then be calculated using the simple formula:

$$Correspondence = \frac{\frac{|A'|}{|A|} + \frac{|B'|}{|B|}}{2}$$

This yields a value within the range 0 to 1. $|A|$ can be re-written as 1+1+1+1+1.... by adding a 1 for every word in A. Each word has already been given a significance value as described in stage 2 of the algorithm and this information is taken into account by re-defining $|A|$ as s1+s2+s3+... where s1 is the significance value assigned to the first word in A, s2 the second and so on. The same can be done for A', B and B'. The formula now takes the average of the biased ratios. All this means is that instead of each word counting for '1' in a set, it counts for its significance value (a value between 0 (insignificant) and 1 (highly significant)). The result is that each word affects the correspondence measure according to its significance in the text.

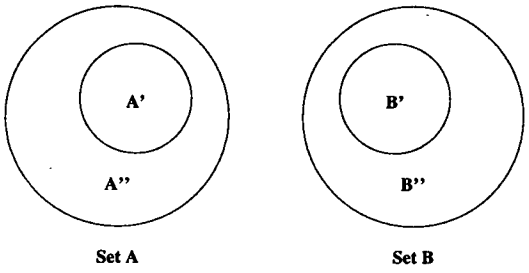

Figure 4: Word Sets.

So far, a word that occurs only in A and not in B, contributes zero to $|A'|$. This means that a highly significant word occurring only in A has exactly the same effect as an insignificant word occurring only in A. In other words the significance biasing is only taking place for words that appear in both A and B. Therefore, the formula actually used is:

$$Correspondence = \frac{\frac{|A'|-|A''|}{|A|} + \frac{|B'|-|B''|}{|B|}}{2}$$

where A'' is the subset of A which contains only those words that occur in A and not in B. Similarly, B'' is the subset of B which contains only those words that occur in B and not in A. This is shown in Figure 4.

Recall that $|A|$, $|A'|$, $|A''|$, $|B|$, $|B'|$ and $|B''|$ are not calculated by adding one for each word in each set, but by summing the significance values of the words in each set.

This stage of the processing looks at the output from the significance calculation stage and considers every sentence break in turn - starting at the top of the document and working down. The algorithm assigns a correspondence measure to each sentence break as follows: Firstly, set A is generated by taking all the words in the previous fifteen sentences. Next, set B is generated by taking all the words in the following fifteen sentences. [1] Now sets A', A'', B' and B'' are generated as described and then the formula above is applied which assigns a correspondence value to the sentence break currently under consideration. The algorithm then moves to the next sentence break and repeats the process.

The output from this stage of the algorithm is a list of sentence break numbers (1..n, with n = number of sentences in the document) and a lexical correspondence measure. These numbers provide the input for stage four - smoothing.

[1] Fifteen sentences turns out to be the optimum window size for the vast majority of texts. This is because it is about the same as the average segment size.

2.4 Smoothing (stage 4)

A graph can be plotted with lexical correspondence along the y-axis and sentence number along the x-axis. In order to distinguish the significant peaks and troughs from the many minor fluctuations, a simple smoothing algorithm is used. Taking three neighbouring points on the graph, P1, P2, P3:

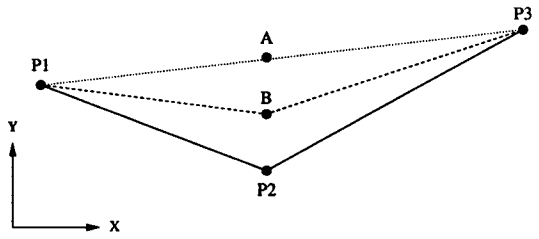

Figure 5: Smoothing.

The line P1P3 is bisected and this point is labelled A. P2 is perturbed by a constant amount (*not* dependent on the distance between A and P2) towards A. This new point is labelled B and becomes the new P2. This is performed simultaneously on every point on the graph. The process is then iterated a fixed number of times. The result is that noise is flattened out while the larger peaks and troughs remain (although slightly smaller).

The output from this stage is simply the sentence break numbers and their new, smoothed correspondence values.

2.5 Inserting subject boundaries (stage 5)

Considering the graph described in the previous section, generating subject boundaries is simply a matter of identifying local minima on the graph. The confidence of a boundary is calculated from the 'depth' of the local minimum. This depth is calculated simply by taking the average of the heights of the 'peak' (relative to the height of the minimum) on either side of the minimum. This now yields a list of candidate subject boundaries and an associated confidence measure for each one. Breaks are then inserted into the original text at the places corresponding to the local minima if their confidence value satisfies a 'minimum confidence' criterion. This cut-off criterion is arbitrary, and in our implementation can be specified at run time.

3 Results

Figure 6 shows the result of processing the first 800 sentences from an edition of The Times newspaper. The sentence number (x-axis) is plotted against the correspondence (y-axis) between the two windows of text on either side of that sentence.

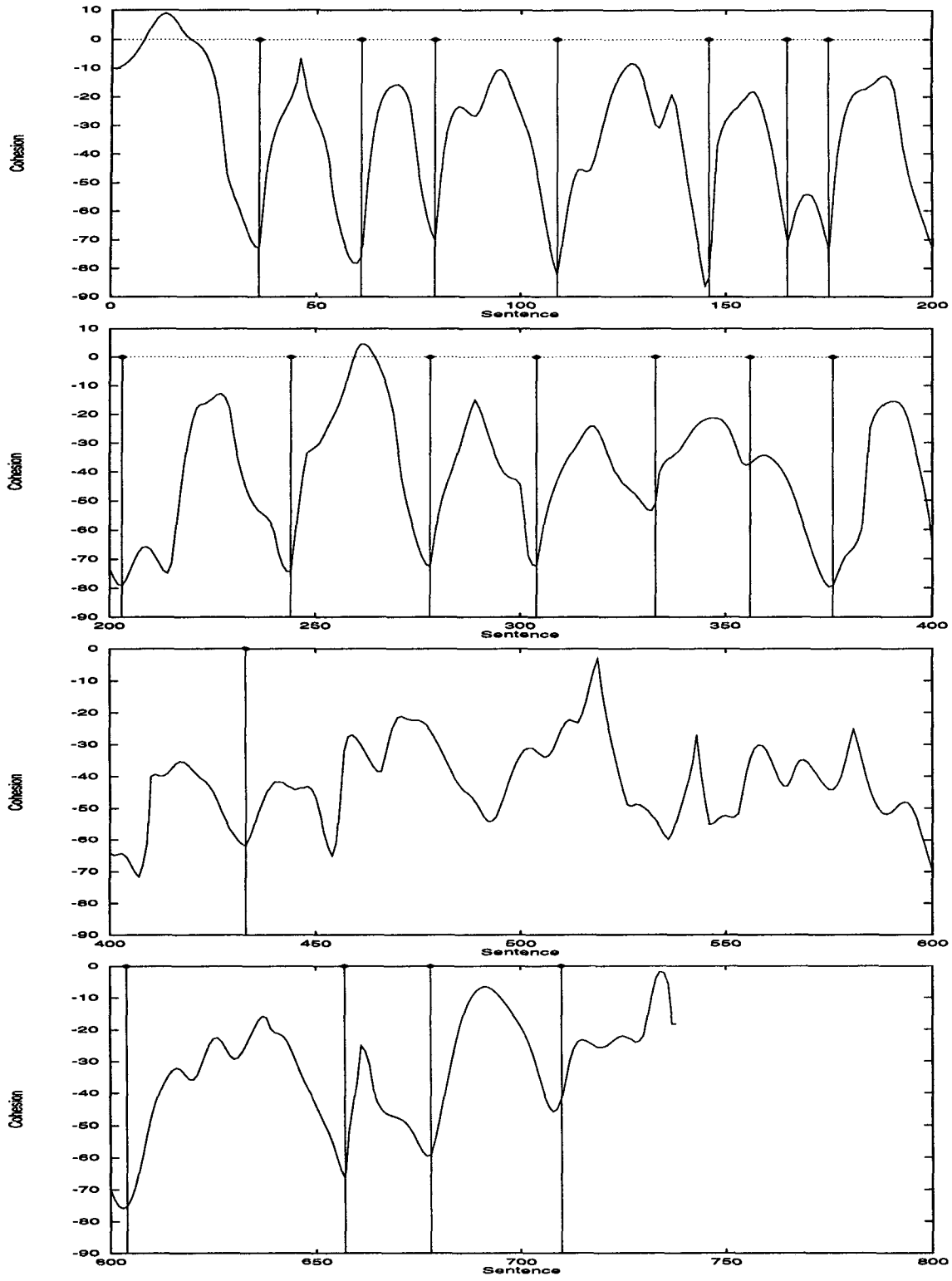

Figure 6: 800 Sentences from The Times newspaper.

Actual subject boundaries	Boundaries found by algorithm	Error
36	36	0
61	60	1
79	79	0
109	109	0
-	134	+
146	145	1
165	165	0
175	174	1
203	203	0
-	214	+
244	244	0
278	278	0
304	304	0
333	332	1
356	355	1
376	375	1

Table 1: The Times

Actual subject boundaries	Boundaries found by algorithm	Error
7	7	0
22	22	0
-	42	+
59	58	1
72	72	0
-	77	+
96	92	4
121	118	3
-	137	+
-	156	+
162	161	1
-	177	+
184	184	0
-	191	+

Table 2: Expository Text

A large negative value indicates a low degree of correspondence and a small negative value or a positive value indicates a high degree of correspondence. The vertical lines mark actual article boundaries.

The advantage of using a text such as this is that there can be no doubt from any human judge as to where the boundaries occur, i.e. between articles. The local minima on the graph signify the boundaries as determined by the algorithm. The vertical bars signify the actual article boundaries. The results of the first 400 sentences are summarised in table 1.

The algorithm located 53% of the article boundaries precisely and 95% of the boundaries to within an accuracy of a single sentence. *Every* article boundary was identified to within an accuracy of two sentences. The algorithm made no use of end-of-paragraph markers. It also found some additional subject boundaries in the middle of articles. These are denoted by a '+' in the error column. Many extra subject boundaries were found in the long article (starting at sentence 430). It is worth noting that the minima occurring within this article are not as pronounced as the actual article boundaries themselves. This section of the graph reflects a long article which contains a number of different subtopics.

A newspaper is an easy test for such an algorithm though. Figure 7 shows a graph for an expository text - a 200 sentence psychology paper written by a fellow student. Again the local minima indicate where the algorithm considers a subject boundary to occur and the vertical lines are the obvious breaks in the text (mainly before new headings) as judged by the author. The results are summarised in table 2.

This time the algorithm precisely located 50% of the boundaries. It found 63% of the boundaries to within an accuracy of a single sentence and 88% to within an accuracy of two sentences. This level of accuracy was obtained consistently for a variety of different texts. Again, it should be mentioned that the algorithm found more breaks than were immediately obvious to a human judge. However, it should be noted that these extra breaks were usually denoted by smaller minima, and on inspection the vast majority of them were in sensible places.

The algorithm has a certain resolving power. As the subject matter becomes more and more homogeneous, the number of subject breaks the algorithm finds decreases. For some texts, this results in very few divisions being made. By taking a smaller window size (the number of sentences to look at either side of each possible sentence break), the resolving power of the algorithm can be increased making it more sensitive to changes in the vocabulary. However, the reliability of the algorithm decreases with the increased resolving power. The default window size is fifteen sentences and this works well for all but the most homogeneous of texts. In this case a window size of around six is more effective. A lower window size increases the resolving power, but decreases the accuracy of the algorithm. The window size was a parameter of our implementation.

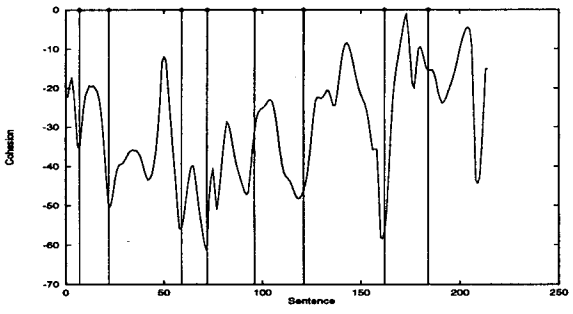

Figure 7: Expository Text.

4 Summary

Based on our investigation, we believe that Hearst's original intuition that lexical correspondences can be exploited to identify subject boundaries is a sound one. The addition of the significance measure represents an improvement on Hearst's algorithm implemented by the Berkeley Digital Library Project.

Furthermore, this algorithm is language independent except for the preprocessing stage (which can be omitted with only a modest degradation in performance). In order to improve accuracy, language dependent methods could be considered. Such methods might include the insertion of conventional discourse markers in order to detect preferred breaking points (e.g. repetition of the same syntactic structure, and conventional paragraph openings such as: "On the other hand...", "The above...", etc.). Another method would be to make use of a thesaurus, since we have found that human judgement is often based on synonymous information such as real synonyms or anaphora. The above issues are discussed in various articles (Morris and Hirst, 1991); (Morris, 1988) and (Givon, 1983) which study discourse markers and synonymous information.

Another interesting line of research would be to use the information from stage two of the algorithm to discover the significant words of a section, and thereby attach a label to it. This would be particularly useful for information retrieval applications.

5 Acknowledgements

This problem was set as an assignment on the Data Intensive Linguistics course organised by Chris Brew at the HCRC, Edinburgh University. Thanks to him, Jo Calder and Marc Moens for guidance and advice throughout the project. Thanks to ESRC and EPSRC for funding.

References

Church, K. W. and W. A. Gale. 1995. Poisson mixtures. *Natural Language Engineering*, 1(2):163–190.

Givon, T. 1983. *Topic Continuity in Discourse: A Quantitative Cross-Language Study.* Philadelphia: John Benjamins Publishing Company.

Hearst, M. A. 1994. Multi-paragraph segmentation of expository text. In *ACL '94*, Las Cruces, NM.

Justeson, J. S. and S. M. Katz. 1995. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1(1):9–27.

Katz, S. M. 1996. Distribution of context words and phrases in text and language modelling. *Natural Language Engineering*, 2(1):15–59.

Morris, J. 1988. Lexical cohesion, the thesaurus, and the structure of text. Technical Report CSRI-219, Computer Systems Research Institute, University of Toronto.

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