TextLec: A Novel Method of Segmentation by Topic Using Lower Windows and Lexical Cohesion

Laritza Hernández Rojas and José E. Medina Pagola

Advanced Technologies Application Centre (CENATAV)
7a #21812 e/ 218 y 222, Rpto. Siboney, Playa. C.P. 12200, C. Habana, Cuba
{lhernandez,jmedina}@cenatav.co.cu

Abstract. The automatic detection of appropriate subtopic boundaries in a document is a difficult and very useful task in text processing. Some methods have tried to solve this problem, several of them have had favorable results, but they have presented some drawbacks as well. Besides, several of these solutions are application domain dependant. In this work we propose a new algorithm which uses a window below the paragraphs to measure the lexical cohesion to detect subtopics in scientific papers. We compare our method against two algorithms that use the lexical cohesion too. In this comparison we notice that our method has a good performance and outperforms the other two algorithms.

Keywords: Text processing, Segmentation by topic, Lexical cohesion.

1 Introduction

The concept of segmentation in text processing has been used with different interpretations. For example, Boshakov and Gelbukh in 2001 structured it as: thematic (topic) and lexico-grammatical segmentation [3]. In this work, we will focus on topic segmentation.

A document usually contains several pieces of text about a more specific content (subtopic) regarding the content of the whole text (topic). Such pieces are formed by textual units (words, sentences or paragraphs). The marks, subtitles or comments that can identify the subtopics are not always used by the authors of the documents. The pieces of text obtained by means of automatic processing, that identify the subtopics that form the text, are known as segments, and this process is known as Texts Segmentation by Topic.

The process of segmentation by topic is useful in several text processing tasks. For example, Texts Summarization, News stories segmentation from broadcast news, Information Retrieval and others [1], [5], [10], [11], [13].

The segmentation methods by topic are used in Information Retrieval specifically in Passages Retrieval to return segments or passages more related with the user's queries instead of the whole document. Text Summarization would be more robust by knowing all the subtopics that form a document, because they can be used as a guide to select the main ideas which may include a summary of the whole document.

L. Rueda, D. Mery, and J. Kittler (Eds.): CIARP 2007, LNCS 4756, pp. 724–733, 2007.

The segmentation that we will propose in this work will be used in a scientific information retrieval system to find document segments that topically satisfy specific requirements or profiles.

A scientific paper is usually a text of multiple paragraphs that explicitly explain or teach about a topic, in which the most significant words that form a subtopic are frequently reiterated.

Although there are some approaches to solve the segmentation problem, the results that they achieve do not always have a high quality.

This and all the reasons exposed here have motivated our work. We propose in this paper a new segmentation algorithm, which is based on the following assumption: if a lexical terms group (vocabulary) is used during the course of the discussion of a subtopic and this subtopic changes, a significant portion of this vocabulary changes too.

We have structured the present work as follows. In Section 2 we briefly explain some previous works to solve the segmentation problem and their drawbacks. In Section 3 we describe the proposed method. In the last section we present the experimental results by using a textual corpus which we prepared with articles selected from the ICPR '2006 proceedings.

2 Methods of Segmentation by Topic

Many of the researches about segmentation by topic use the linguistic term Lexical Cohesion. This term was defined by Halliday and Hasan in 1976 as a sense relationship that exists among the textual units in a text [4]. The repetition or lexical reiteration, synonymy and others are mechanisms that indicate sense relationships.

The results of these researches have shown that the Lexical Cohesion is a very useful way to detect the subtopics changes inside a text, because the textual units that are strongly related by Lexical Cohesion usually constitute a segment about a simple subtopic [5], [6],[9],[12].

We consider proper to mention that there are some works in segmentation focused on discovery of topical units and not on subtopic structure inside documents although this is not the goal of this work.

The Stokes, Carthy and Smeaton's work is an example of the previous one. This work is called SeLeCT, which is intended to distinguish individual news stories from a broadcast news programme [12]. SeLeCT is based on an analysis of the lexical cohesion strength among the textual units using a linguistic technique that is called Lexical Chaining [13].

Ponte and Croft proposed a method that has as application goal the topic tracking of broadcast speech data and topic identification in full-text databases. Their work is focused on text with relatively small segment sizes and for which sentences inside a segment have relatively few words in common turning segmentation into a more difficult problem. This method uses a query expansion technique to find common features for the topic segments [8].

Next we will describe two segmentation methods that are focused on the identification of subtopic structures in documents; they are based on lexical reiteration to detect relationship among textual units.

2.1 Segmentation Proposed by Hearst

We found that the most interesting research in identification of subtopic structures is the Hearst work. She proposed an algorithm which he called TextTiling. This algorithm splits explanatory texts into discourse units of multiple paragraphs. In contrast to many discourse models that assume a hierarchical segmentation model, the author has the goal of identifying subtopic boundaries by attempting only a linear segmentation. Also, Hearst assumes that if a lexical terms group (vocabulary) is used during the course of the discussion of a subtopic and this subtopic changes, then a significant portion of this vocabulary changes too.

The algorithm has three main parts: pre-processing, lexical score determination and boundary identification.

In the first one stopwords are eliminated and a morphological analysis is applied to the text. Besides, the text is subdivided into sequences of a predefined size of the resulting words, without considering punctuation marks, which she called sentences (pseudosentences).

Then a lexical score is determined. TextTiling proposes two lexical score methods. The first method compares adjacent blocks of sentences, and assigns a similarity score between two blocks according to how many words they have in common, the sentences blocks are represented by the Vector Space Model. The second method, called vocabulary introduction, forms text intervals with sentences and assigns a lexical score to the midpoint of the interval, based on how many new words (words not seen before in the text) appear around this midpoint.

Finally, the identification of the limit is made identically for the two lexical scoring methods. Keeping in mind this lexical score, a depth score is assigned to each gap between blocks with lowest lexical scores, called valleys.

The depth score of a valley corresponds to how strongly the features for a subtopic changed on both sides of the valley and is based on the distance from the valley to the two peaks that make it up. In other words, if a low lexical score is preceded and is followed by a high lexical score this is assumed as an indicator of a change in the vocabulary, which will correspond, according to what the author assumed, to a subtopic change.

Then, the depth scores are sorted and used to determine segment boundaries. The larger the score, the more likely the boundary occurs at that location [5].

This algorithm maintains a good performance, but it presents a drawback that causes the interruption of a segment that contains a simple subtopic; this problem also produces many segments that surpass the considered valid amount. This occurs when there is a short paragraph or other (e.g. paraphrase) which interrupts a cohesive text chain. TextTiling does not detect this behavior. In these anomalous cases, TextTiling gets a notable low score and, then, assigns a segment boundary.

2.2 Segmentation Proposed by Heinone

Unlike Hearst, Heinone proposed a method which uses a sliding window to determine, for each paragraph, which is the most similar paragraph inside the window. The sliding window will be formed by several paragraphs on both sides (above and below) of every processed paragraph.

This segmentation method is especially useful when it is important to control the segment length. The author uses a dynamic programming technique that guarantees to get segments of minimum cost. The segment cost is obtained by a lexical cohesion curve among the paragraphs, a preferential segment size specified by the user, and a defined parametric cost function [6].

Firstly, as in TextTiling, Heinone suggests a text preprocessing stage. Then, the paragraphs are represented by the Vector Space Model similar to TextTiling as well.

Later, a cohesion vector ($Cohe_1 ... Cohe_n$) is built for the document, where each paragraph is associated with the highest similarity value inside its window.

As the algorithm considered the segment length, a length cost function $clen(x, p, h)$ is used to evaluate the closeness between the real and the preferential segment length, where x is the real segment length, p the preferential segment length, and h a scale parameter to adjust the length.

The algorithm calculates, in a sequential way from the first to the last one, the minimum cost segmentation by paragraph. To this end, the following expressions were considered:

$$Cost_i = \min(CostS(S_i^i) ... CostS(S_i^1)) \ , \tag{1}$$

$$Cost_0 = 0 \ , \ Cohe_0 = 0 \ , \tag{2}$$

$$CostS(S_i^k) = Flon(S_i^k) + Cohe_{k-1} + Cost_{k-1} \ , \tag{3}$$

$$Flon(S_i^k) = clen(\text{"amount of words in segment from k to i"}, p, h) \ , \tag{4}$$

were S_i^k is a segmentation that considers a segment from k-th to i-th paragraphs.

Besides, for each paragraph, its segment boundary is defined, which is the last paragraph of the previous segment. This boundary is determined by the expression:

$$LimP_i = k - 1 \ \ were \ \ Cost_i = CostS(S_i^k) \ . \tag{5}$$

This method achieves a good assessment among the segment length, the preference length and the similarity value associated with each paragraph. Besides, the use of a sliding window can diminish the interruption of a cohesive text chain, in contradistinction to Hearst's proposal.

However, this method entails a shortcoming as well. The cohesion vector associates each paragraph with the highest similarity value in its window, but this value can belong to a paragraph above or below the paragraph in question. In case the value belongs to a paragraph above, the algorithm – not distinguishing this situation – could decide to include the paragraph in the segment below. As we can observe, it is incorrect to allow the similarity value to be chosen with paragraphs above to decide the inclusion of a paragraph in a segment below. This weakens the method assumptions by enabling low cohesion segments to be obtained.

On the other hand, for each paragraph, the method looks for the most similar paragraph inside its window, causing that other very similar paragraphs are underestimated.

Besides, this method has another drawback; it requires an approximate subtopic length, which is an unpredictable value that is not always the same for all the subtopics. Lastly, when the cost length function establishes a matching between the approximate and real length, the algorithm can interrupt the segment cohesion.

3 Segmentation Method TextLec

We propose a novel method called TextLec, which intends to segment scientific papers into subtopics. With TextLec we solve the drawbacks aforementioned. We also use term repetitions as a mechanism to indicate the lexical cohesion among the text units.

In this work we take the paragraph as the minimum text unit, because, as we have said before, our work is intended to the segmentation of multiple paragraphs.

We assume a lineal segmentation and also that whenever there is a subtopic change we are in the presence of a vocabulary change, just as Hearst assumed. Besides, we consider that: the neighbor paragraphs that maintain a considerable lexical cohesion among them, regarding the use of a lexical terms group, should belong to the same segment.

We establish that, if the value that expresses the lexical cohesion between two paragraphs is bigger than a similarity threshold U, then these paragraphs have a considerable lexical cohesion among them. From now on, we only use the term *cohesive* to refer to paragraphs whose lexical cohesion is bigger than the threshold U.

With the frequency of the resulting terms after a preprocessing, we represent the paragraphs with the Vector Space Model. We will compute the lexical cohesion by applying the cosine measure.

For each paragraph, we define a lower window to find inside a cohesive paragraph that will be, opposite to Heinone, the farthest cohesive paragraph.

We only use a lower window to distinct if paragraphs, above or below, are cohesive, and to diminish the possibility of wrongly including a paragraph into a segment. Besides, we decided to take the farthest cohesive paragraph inside each window instead of the most cohesive; so, we do not leave all the potential cohesive paragraphs, what diminishes the possibility of excluding a paragraph incorrectly from a segment. This also allows us to avoid more efficiently the effect of paragraphs that interrupt a coherent text chain.

Besides, with the control of the farthest cohesive paragraph we can determine a possible end of a segment.

Another interesting aspect of this method is that by increasing the window size we are able to obtain longer segments, because we increase the possibility of finding a farthest cohesive paragraph, although the segment cohesion would decrease.

3.1 Preliminary Conditions

We use the vector *Parf* with the purpose of controlling the farthest cohesive paragraph inside the window for each paragraph. It is possible that a paragraph does not have any cohesive one inside its window; in this case, we consider that the paragraph is cohesive at least to itself.

On the other hand, we use the variable *MaxBel* to control the possible end (lower boundary) of the segment in process while the segmentation algorithm is executed. This variable will allow knowing the farthest cohesive paragraph from the segment in process. Therefore, the paragraph controlled by *MaxBel* will be possibly the one that closes the segment.

Besides, we use the vector *Lim* to control, for each segment, the lower boundary, where Lim_k will contain the finish paragraph of segment *k*. For example, if $Lim_k = 5$, then we will get a segment *k* that finishes in paragraph 5.

3.2 Segmentation Process

The segmentation process that we propose will include paragraphs into a segment until finding a paragraph that is not cohesive with any paragraph inside the segment and also if there is no paragraph inside the segment cohesive with a paragraph below it.

We show the segmentation algorithm pseudo-code in Fig. 1.

```
Algorithm: TextLec
Input:  Parf – vector of more similar paragraph below
        N – total of paragraphs
Output: Lim – boundary of segments
 1) MaxBel = Parf1;
 2) k = 1;
 3) for i = 2 to N do begin
 4)   if MaxBel = i-1; then begin
 5)     Limk = MaxBel;
 6)     k = k + 1;
 7)   end
 8)   MaxBel = max( Parfi, MaxBel );
 9) end
10) Limk = N
```

Fig. 1. Pseudo-code of TextLec Algorithm

We suppose that the texts will have one paragraph at least, and we also suppose that we will also obtain a segment formed by one paragraph at least. Then we have as initial values: $MaxBel = Parf_1$ and $k = 1$.

During the execution, we analyze the other paragraphs and we determine whether we include a paragraph or not in the segment that we are building. During this process, a paragraph *i* can be in any of the following situations:

- There is no paragraph inside the segment that is cohesive with *i* or cohesive with a paragraph below *i*, ($MaxBel = i - 1$). In this case we take the paragraph $i-1$ as lower boundary of the segment, and we include *i* into a new segment.
- The paragraph *i* is cohesive with some paragraph inside the segment, ($MaxBel = i$). In this case we include *i* into this segment.
- There is no a paragraph inside the segment, that is cohesive with *i*, but there is at least one inside cohesive with a paragraph below *i*. Then, we do not interrupt this segment and we include *i* into it.

After the inclusion of the paragraph *i* into a segment we verify if the cohesive paragraph with *i* is farthest than *MaxBel*; in this case, we update *MaxBel*.

When the process finishes all the lower boundary of the segments are into *Lim*, and *k* is the amount of detected segments.

4 Evaluation

The evaluation of the segmentation by topic has two major related difficulties. The first one is given by the subjective nature of detecting the appropriate subtopic boundaries, in which several human readers can even disagree, regarding where boundaries should be placed and how fine-grained an analysis should be. This makes it difficult for us to select a reference segmentation to compare our results. Usually, this difficulty is solved by comparing the segmentation results against the marks or subtitle that the authors use to identify the subtopics inside a document; but these marks are not always specified. Some authors evaluate the algorithm in terms of how well it distinguishes entire articles from one another when they are concatenated into one file and where different topics are distinguished. Another way is to compare the results against a manual segmentation based on the several human judgments, which make a "gold standard" [5], [7], [12].

The second difficulty is that error importance depends on the applications where the segmentation techniques are necessary. For example, in Information Retrieval it can be accepted that the segment boundaries will differ in a few sentences from the real segment boundaries. However, in order to segment news stories from broadcast news the accuracy of boundaries location is very important [7].

On the other hand, finding an appropriate evaluation metric to determine the segmentation algorithm accuracy has generated much debate. Two of the evaluation measures that have been used by many authors are Precision and Recall, which are standard measures in Information Retrieval experimentations. In the estimation of the segmentation accuracy, the Precision and the Recall are defined likes this.

Precision: The percentage that represents the segment boundaries correctly detected by the algorithm from all boundaries detected by it.

Recall: The percentage that represents the segment boundaries correctly detected by the algorithm from all boundaries in the reference segmentation.

Precision and Recall are usually very convenient in applications where the accuracy of boundaries location is very important. But in applications where it is not very necessary they have some problems. These measures strongly penalize the algorithm when boundaries that do not agree exactly with the reference segmentation are detected, because they are not sensitive to the proximity between the boundaries of both segmentations. Another difficulty with Precision and Recall is that there is inherent tradeoff between precision and recall; improving one tends to cause the score for the other to decline [2], [7]. This difficulty is usually solved in Information Retrieval with F-measure; it is defined as:

$$\text{F-measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \ . \tag{7}$$

F-measure has been used in segmentation as well. Nevertheless, we should note that as F-measure depends on Precision and Recall it shows the first problem, i.e., it is not sensitive to the proximity between the boundaries of both segmentations.

Pevzner and Hearst proposed a metric to improve the segmentation evaluation process, called WindowDiff [7]. WindowDiff uses a sliding window of length k to find disagreements between the reference and algorithm segmentation.

In the segmentation literature there are many authors that experiment with several window sizes, i.e., with several k values. In this work we take k as the half of the average true segment size in the reference segmentation.

The boundaries amount inside the window of both segmentations is determined for each window position; the algorithm is penalized if the amount of boundaries disagrees. Later, all penalizations found are added. This value is normalized and the metric takes a value between 0 and 1. WindowDiff takes a score of 0 if the algorithm assigns all boundaries correctly and it takes a score of 1 if it differs completely from the reference segmentation. The WindowDiff formal expression is defined as follows:

$$WindowDiff(ref, hyp) = \frac{1}{N-k}\sum_{i=1}^{N-k}(|\, b(ref_i, ref_{i+k}) - b(hyp_i, hyp_{i+k})\,| > 0)\ , \tag{8}$$

where $b(i, j)$ represents the number of boundaries between positions i and j in the text and N represents the textual units number in the whole text, regarding goal applications; ref is the reference segmentation and hyp the algorithm segmentation.

In this section we show the result of the three segmentation methods: TextTiling, the Heinone's, and TextLec. The corpus that we used in the experimentation was built joining 14 different papers taken from The 18th International Conference on Pattern Recognition ICPR'2006 proceedings. The resultant corpus has 305 paragraphs and an average of 22 paragraphs approximately for each paper.

We chose from all the marks or subtitles inside the papers defended by the authors those that, in our opinion, have the clearest boundaries to select the reference segmentation based on our judgment. Also, we added to the reference segmentation the boundaries among each different article.

We show the comparison among the three algorithms by calculating the Precision, Recall and metric WindowDiff values for each one, regarding the reference segmentation. These values are shown in Table 1.

Table 1. Precision, Recall and metric WindowDif values for three segmentation methods: TextLec TextTiling, and the Heinone's

Algorithms	Precision	Recall	F-measure	WindowDiff
TextLec	61,74	52,59	56,8	0,21
TextTiling	45,90	62,22	52,83	0,33
Heinone's	45,24	42,2	43,67	0,26

In the experimentation we can notice a better performance in TecxLec compared to the other two algorithms. TextLec obtained a better WindowDiff value, i.e., closer to 0. Also, both values, Precision and Recall, are greater than 50%, but we should notice that if the amount of obtained boundaries increases then the Recall may improve but,

at the same time, Precision may decline. For example, TextTiling has a recall value higher than TextLec, but it has a lower precision value and lower than 50%. In any case, TextLec has a higher F-measure value than the TextTiling and Heinone's algorithm.

5 Conclusion

The use of text methods of segmentation by topic would improve the results of many text processing tasks; for example, Text Summarization, News stories segmentation from broadcast news, Information Retrieval, and others. We proposed here a new segmentation algorithm, which has as a goal the segmentation of scientific papers to apply its results on a scientific information retrieval system.

We based our research on term repetition as a lexical cohesion mechanism to determine strongly cohesive segments. The paragraphs were represented by the Vector Space Model. We measure the lexical cohesion by computing the similarity among the paragraphs, and using the cosine measure. In this way we could diminish the possibility of interrupting the subtopic coherence obtaining more cohesive segments, what increases the TextLec performance when we compare it with other segmentation algorithms.

As future work, we propose to continue improving this method looking for other computational models to represent the textual units, and determining with greater robustness the lexical cohesion between these textual units, to detect the subtopics in a document with better precision.

References

1. Angheluta, R., Busser, R., Moens, M.F.: The Use of Topic Segmentation for Automatic Summarization. In: Proceedings of the ACL-2002, Post-Conference Workshop on Automatic Summarization (2002)
2. Beeferman, D., Berger, A., Lafferty, J.: Text segmentation using exponential models. In: Proceedings of the Second Conference on Empirical Methods in Natural Language Processing, pp. 35–46 (1997)
3. Bolshakov, I.A., Gelbukh, A.: Text segmentation into paragraphs based on local text cohesion. In: Proceedings of the 4th International Conference on Text, Speech and Dialogue, Lecture Notes in Artificial Intelligence, pp. 158–166 (2001)
4. Halliday, M.A.K., Hasan, R.: Cohesion in English. Longman Group, New York (1976)
5. Hearst, M.A.: TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages. In: Computational Linguistics, vol. 23(1) (1997)
6. Heinonen, O.: Optimal Multi-Paragraph Text Segmentation by Dynamic Programming. In: Proceedings of COLING-ACL 1998, Montreal, Canada, Cite as: arXiv:cs/9812005v1 [cs.CL] pp. 1484–1486 (1998)
7. Pevzner, L., Hearst, M.A.: A Critique and Improvement of an Evaluation Metric for Text Segmentation. In: Computational Linguistics, vol. 16(1) (2000)
8. Ponte, J.M., Bruce Croft, W.: Text segmentation by topic. In: Peters, C., Thanos, C. (eds.) ECDL 1997. LNCS, vol. 1324, pp. 113–125. Springer, Heidelberg (1997)

9. Reynar, J.C.: An automatic method of finding topic boundaries. In: Proceedings of the 32nd annual meeting on Association for Computational Linguistics, Las Cruces, New Mexico, Student session, Cite as arXiv:cmp-lg/9406017 v1 7 Jun 1994 cmp-lg/9406017 7 Jun 1994 (1994) pp. 331–333 (1994)
10. Reynar, J.C.: Topic Segmentation: Algorithms and Applications. Thesis Doctoral, Presented to the Faculties of the University of Pennsylvania, Pennsylvania (1998)
11. Reynar, J.C.: Statistical Models for Topic Segmentation. In: Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics, pp. 357–364 (1999) ISBN:1-55860-609-3
12. Stokes, N., Carthy, J., Smeaton, A-F.: SeLeCT: A Lexical Cohesion Based News Story Segmentation System. AI Communications 17(1), 3–12 (2004)
13. Stokes, N.: Applications of Lexical Cohesion Analysis in the Topic Detection and Tracking Domain. Thesis Doctoral, Department of Computer Science Faculty of Science, National University of Ireland, Dublin (2004)