Text Segmentation by Clustering Cohesion

Raúl Abella Pérez and José Eladio Medina Pagola

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

Abstract. An automatic linear text segmentation in order to detect the best topic boundaries is a difficult and very useful task in many text processing systems. Some methods have tried to solve this problem with reasonable results, but they present some drawbacks as well. In this work, we propose a new method, called ClustSeg, based on a predefined window and a clustering algorithm to decide the topic cohesion. We compare our proposal against the best known methods, with a better performance against these algorithms.

1 Introduction

Text segmentation is the task of splitting a document into syntactical units (paragraphs, sentences, words, etc.) or semantic blocks, usually based on topics. The difficulty of text segmentation mainly depends on the characteristics of documents which will be segmented (i.e. scientific texts, news, etc.) and the segmentation outputs (e.g. topics, paragraphs, sentences, etc.). There are different approaches to solve this problem; one is a linear segmentation, where the document is split into a linear sequence of adjacent segments. Another approach is a hierarchical segmentation; the outputs of these algorithms try to identify the document structure, usually chapters and multiple levels of sub-chapters [6].

There are many applications for text segmentation. Many tools for automatic text indexing and information retrieval can be improved by a text segmentation process. For example, when segmenting news of broadcast story transcriptions, a topic segmentation takes a crucial role, because a topic segmentation can be used for retrieving passages more linked to the query made by the user, instead of the full document [9], [11]. In tasks of summary generation, text segmentation by topics can be used to select blocks of texts containing the main ideas for the summary requested [9].

Analyzing the performance of different methods of text segmentation by topics [7], [8], [9], [10] we observed some difficulties as, for instance, wrong interruptions of segments, leaving out sentences or paragraphs which belong to the segments, and generating segments with incomplete information. When these situations happen, spurious segments are obtained. Another difficulty we observed is that those methods are not able to identify the true relations amongst paragraphs of each segment considering natural topic cohesion.

In this work we propose an algorithm for linear text segmentation of multi-paragraphs based on topics, called ClustSeg, defined as a solution of the aforementioned difficulties. This method is based on a window approach to identify boundaries of

I. Bloch and R.M. Cesar, Jr. (Eds.): CIARP 2010, LNCS 6419, pp. 261–268, 2010.

topics. Each paragraph is represented using the vector space model, similar to other methods [6], [7], [9]. But, unlike these methods, we assumed that the paragraph cohesion is obtained from a clustering method, generating segments containing paragraphs from one topic or a mixture of them.

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 Related Work

The study of the linear text segmentation methods by topics must be initiated by the TextTilling algorithm. This method was proposed by Hearst and it is considered one of the most interesting and complete studies on the identification of structures of sub-topics [7]. In this paper, Hearst proposed a method which tries to split texts into discourse units of multiple paragraphs. This algorithm uses a sliding window approach and for each position two blocks are built; one preceding and the second succeeding each position. To determine the lexical punctuation between these two blocks, it uses the term repetition as a lexical cohesion mechanism. These blocks are formed by a specified amount of pseudo-sentences which are represented by the vector space model and the cosine as the similarity measure. Considering the lexical values calculated, this method splits the text from the valleys, or points with low lexical scores.

Unlike Hearst, Heinone proposed a method which uses a sliding window to determine, for each paragraph, which the most similar paragraph inside the window is [8]. The sliding window is 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 getting 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.

Another approach of linear text segmentation by topic is TextLec, proposed in 2007 [9]. This method, like TextTiling, uses word repetition as a lexical cohesion mechanism. Each paragraph is represented by the vector space model, as in Hearst's and Heinonen's work. The authors of this work assume that all the sentences which belong to a paragraph are about a same topic. This method also uses a sliding window approach but, unlike Hearst, it only uses a window of paragraphs which are below each position. This method consists of two stages; the first finds for each paragraph the farthest cohesive one within the window, using the cosine measure and a threshold. Finally, it searches the segment boundaries in a sequential process, in which a paragraph is included in a segment if it is the farthest cohesive one related to any other paragraph previously included in the segment.

The last method we have considered is the C99 algorithm proposed by Choi [3]. This method is strongly based on the lexical cohesion principle. C99 uses a similarity matrix of the text sentences. First projected in a word vector space representation,

sentences are then compared using the cosine similarity measure. More recently, Choi improved C99 by using the Latent Semantic Analysis (LSA) achievements to reduce the size of the word vector space [4].

3 ClustSeg: A Method for Text Segmentation

In our approach, we assume a linear segmentation and consider paragraphs as the minimum text units. But, in spite of other methods, we assume that sentences belonging to a paragraph, and each paragraph *per se*, could be on several topics. Paragraph representation is based on the vector space model and the topic cohesion is calculated using the cosine measure. Nevertheless, the segment boundaries are defined from the results of a clustering algorithm, as we can see below.

The ClustSeg algorithm begins preprocessing the document. In this stage, the stop-words (prepositions, conjunctions, articles, pronouns) are eliminated, considering that these words are devoid of information to decide how similar paragraphs are amongst them. All punctuation marks, numbers and special characters are also removed. Next, the terms can be transformed by their lemmas. We have used in this work the TreeTager, which is a system with the possibility of extracting the lemmas in various languages.

After this preprocessing, the algorithm continues with three stages: the search for topic cohesion, the detection of topic segment boundaries, and the detection of the document segment boundaries.

3.1 Searching for Topic Cohesion

Although we consider the vector space model and the cosine measure, assuming that a vocabulary change could produce a topic change and also a beginning of a new segment, we do not take that model and measure directly to decide the segment boundaries, as the aforementioned methods do.

The main drawback that all those methods present is the assumption that if two paragraphs or textual units have a high similarity value, then there is a high confidence that these paragraphs belong to a significant topic and could belong to a same segment. We have observed that this assumption does not identify all the spurious segments and, also, it can create segments with paragraphs of weak topic cohesion.

Besides, all the methods we have analyzed assume that each paragraph or textual unit is about a unique topic, and the cosine similarity amongst paragraphs can link them, making up segments based on a topic-driven process. In this paper, we have considered that a paragraph could be about several topics, and the significant cohesion amongst them depends on the related topic and on its significance to the document.

Taking into consideration those drawbacks and hypotheses, we weigh up a clustering process applied to the set of paragraphs, without taking into account their order, as a way to obtain clusters of high topic cohesion. Besides, because of the hypothesis of multi-topic paragraphs, a convenient (or maybe necessary) restriction is that the clustering algorithm could produce overlapped clusters.

There are not so many clustering algorithms oriented to obtain cohesive and overlapped clusters. Examples of them are Star [1], Strong Compact (FCI) [14] and

ICSD [12]. We decided to use the static version of ICSD, considering a good performance (or better according to the authors) on overlapped clusters for topic discovery.

The selected algorithm, called (Incremental) Clustering by Strength Decision (ICSD), obtains a set of dense and overlapped clusters using a graph cover heuristic. It is applied over a thresholded similarity graph formed by the objects (paragraphs represented by the vector space model) as the vertices and the similarity values (using the cosine measure) amongst them. The threshold (defined by the user) is used to determining whether two paragraphs are similar.

So, when the static version of ICSD is applied, we can obtain a set of clusters, where each cluster represents a set of cohesive paragraphs belonging (presumably) to an independent topic. The paragraphs identified by the clustering are the only ones we will consider in the following stages.

3.2 Detecting the Topic Segment Boundaries

After obtaining the clustering from the static version of ICSD algorithm, we process every cluster in order to obtain all the segments which can be formed from each cluster.

As in other methods, we use a window (W) to define if two adjacent paragraphs in a cluster are close. Each segment is obtained linking adjacent paragraphs according to the predefined window.

The result of this stage is a set of segments defined by a set of pairs $\langle I, F\rangle$, where I and F represent the indexes of the initial and final paragraphs of the segment. The algorithm of this stage is shown in fig. 1.

```
Algorithm: Topic Segmentation
Input:  C - Clustering of paragraph indexes;
        W - A predefined window;
Output: SI - Set of ⟨I, F⟩ obtained from C;
 1) for each c ∈ C do begin
 2)    j = 1;
 3)    c' = {p_1, …, p_k} by sorting c in ascending order;
 4)    while j < k do begin
 5)       I = First p_i ∈ c' / p_{i+1} - p_i ≤ W and i ≥ j;
 6)       if does not exist I then j = k
 7)       else begin
 8)          F = p_{i+1};
 9)          j = i+2;
10)          while p_j - p_{j-1} ≤ W do begin
11)             F = p_j;
12)             j = j+1;
13)          end
14)          SI = SI ∪ {⟨I, F⟩};
15)       end
16)    end
17) end
```

Fig. 1. Topic segment boundaries detection algorithm

3.3 Detecting the Document Segment Boundaries

As a result from the previous stage, we have obtained a set of topic segment boundaries. As these segments were obtained from different clusters, and they are overlapped, these segments could be also overlapped.

In this stage, we concatenate all the segments that have at least one common paragraph. Observe that, with this consideration, we obtain a linear segmentation satisfying the following condition: A document segment could be made up by a concatenation of a set of topic segments from different topics only if any topic segment has a non null intersection (at least one paragraph) with another one. The Fig. 2 shows this third stage.

Algorithm: Document Segmentation	
Input: *SI* – Set of ⟨*I*, *F*⟩ from *Topic Segmentation*; **Output**: *SF* – Set of ⟨*I*, *F*⟩ obtained from *SI*;	
1)	**for each** ⟨*I*, *F*⟩ = $\underset{\langle I', F' \rangle}{\text{MinArg}}$ {*I'* / ⟨*I'*, *F'*⟩ ∈ *SI* } **do begin**
2)	⟨*In*, *Fn*⟩ = ⟨*I*, *F*⟩;
3)	*SI* = *SI* \ {⟨*I*, *F*⟩};
4)	**while exist** ⟨*I1*, *F1*⟩ ∈ *SI* **and** *I1* ≥ *I* and *I1* ≤ *Fn* **do begin**
5)	*Fn* = max(*Fn*, *F1*);
6)	*SI* = *SI* \ {⟨*I1*, *F1*⟩};
7)	**End**
8)	*SF* = *SF* ∪ {⟨*In*, *Fn*⟩};
9)	**End**

Fig. 2. Document segment boundaries detection algorithm

4 Evaluation

There are two main problems related to evaluation of text segmentation algorithms. The first one is given by the subjective nature when detecting the right boundaries of topics and sub-topics into texts; it turns the selection of reference segmentation for a fair and objective comparison into a very difficult task [13]. In order to solve this problem, usually artificial documents are created, concatenating different real documents, on the assumption that the limits between these documents are good breaking points [6], [9], [13]. Another way is to compare the results against a manual segmentation based on human judgments, which makes a "gold standard" [18].

The second problem is the selection of a measure to use in the evaluation; because, for different applications of text segmentation, different types of mistakes become more or less important. For example, in information retrieval, segment boundaries that differ from the real ones in some few sentences can be accepted. For evaluating a method with this goal, measures like Precision or Recall should not be used. However, when segmenting news of broadcast stories transcription, the accuracy of the boundaries is very important.

In our proposal, the accuracy of the boundaries is not important, because we are trying to automatically discover topic segmentations. One of the best measures to evaluate this task is WindowDiff, a measure proposed by Pevzner and Hearst in 2000 [13].

The WindowDiff measure uses a sliding window of length k to find disagreements between the reference and the algorithm segmentation. In this work we take k as the half of the average true segment size in the reference segmentation.

The amount of boundaries inside the window of both segmentations is determined for each window position; it 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 all boundaries are correctly assigned and it takes a score of 1 if there is a complete difference between the automatic segmentation and the reference one. The WindowDiff formal expression and other details of this measure can be seen in Pevzner and Hearst [13].

In this section we show the results of five segmentation algorithms: ClustSeg, TextLec, TextTiling, Heinone's and C99. The corpus that we used in the experimentation is the same as Hernandez&Medina's work [9]. This corpus was built joining 14 different papers taken from the ICPR'2006 proceedings. The resultant corpus has 305 paragraphs and an average of 22 paragraphs approximately for each paper. We took the segmentation output of our algorithm and we compared it with the results shown in Table 1 of Hernandez&Medina's work [9]. Besides, we included the C99 algorithm results. The C99 algorithm used was downloaded on May 26th from the following *url:http://sourceforge.net/projects/textsegfault/files/c99/C99-1.2-release.tgz/download.*

In Table 1 we can see the results of this experimentation. It was done with a window (W) equal to 10 and a threshold of 0.35 for deciding a significant cosine value between two paragraphs. We can notice a significantly better performance of ClustSeg. We also did other evaluations, varying the window size and the threshold, but these results can not be included here because of the page restrictions. Nevertheless, the ClustSeg algorithm achieved better results than the others for windows size in [9, 11] and threshold in [0.25, 0.38], with a best result of 0.11 for a windows of 11 and a threshold of 0.27 and 0.28.

Table 1. WindowDiff values

Algorithms	ClustSeg	TextLec	TextTilling	Heinone's	C99
WindowDiff	0.12	0.21	0.33	0.26	0.21

Although we did not accomplish any experimentation in terms of execution time, we ran our algorithm and measured the time consumption. The execution time was 1124 ms. Besides, the execution order of this algorithm is $O(m \times n \times \log_2 n)$, including the clustering stage, where m is the amount of clusters and n the amount of paragraphs.

5 Conclusion

The use of text methods of segmentation by topic would improve the results of many text processing tasks; for example, text summarization, information retrieval and others. We have proposed a new segmentation method for discovering topic boundaries.

Although we use the vector space model and the cosine measure, we consider that the paragraph cohesion can be better obtained from a clustering method, generating segments containing paragraphs from one topic or a mixture of them.

The ClustSeg algorithm was compared with four methods, obtaining more cohesive segments and increasing significantly its performance.

As future work, we propose to evaluate other clustering methods and to achieve a better integration of both strategies.

References

1. Aslam, J., Pelekhov, E., Rus, D.: The star clustering algorithm for static and dynamic information organization. Journal of Graph Algorithms and Applications 8(1), 95–129 (2004)
2. Beeferman, D., Berger, A., Lafferty, J.: Statistical Models for Text Segmentation. In: Second Conference on Empirical Methods in Natural Language Processing, pp. 35–46 (1997)
3. Choi, F.Y.Y.: Advances in domain independent linear text segmentation. In: NAACL 2000, pp. 26–33 (2000)
4. Choi, F.Y.Y., Wiemer-Hastings, P., Moore, J.: Latent semantic analysis for text segmentation. In: EMNLP, pp. 109–117 (2001)
5. Filippova, K., Strube, M.: Using Linguistically Motivated Features for Paragraph Boundary Identification. In: The 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pp. 267–274 (2006)
6. Ken, R., Granitzer, M.: Efficient Linear Text Segmentation Based on Information Retrieval Techniques. In: MEDES 2009, Lyon, France (2009)
7. Hearst, M.: TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages. Computational Linguistics 23(1), 33–64 (1997)
8. Heinonen, O.: Optimal Multi-Paragraph Text Segmentation by Dynamic Programming. In: COLING-ACL 1998, Montreal, Canada, pp. 1484–1486 (1998)
9. Hernández, L., Medina, J.: TextLec: A Novel Method of Segmentation by Topic Using Lower Windows and Lexical Cohesion. In: Rueda, L., Mery, D., Kittler, J. (eds.) CIARP 2007. LNCS, vol. 4756, pp. 724–733. Springer, Heidelberg (2007)
10. Labadié, A., Prince, V.: Finding text boundaries and finding topic boundaries: two different tasks. In: Nordström, B., Ranta, A. (eds.) GoTAL 2008. LNCS (LNAI), vol. 5221, pp. 260–271. Springer, Heidelberg (2008)
11. Misra, H., et al.: Text Segmentation via Topic Modeling: An Analytical Study, Hong Kong, China (2009)
12. Pérez-Suáerez, A., Martínez, J.F., Carrasco-Ochoa, J.A.: A New Incremental Algorithm for Overlapped Clustering. In: Bayro-Corrochano, E., Eklundh, J.-O. (eds.) CIARP 2009. LNCS, vol. 5856, pp. 497–504. Springer, Heidelberg (2009)
13. Pevzner, L., Hearst, M.: A Critique and Improvement of an Evaluation Metric for Text Segmentation. Computational Linguistics 28(1), 19–36 (2002)

14. Pons-Porrata, A., Ruiz-Shulcloper, J., Berlanga-Llavori, R., Santiesteban-Alganza, Y.: Un algoritmo incremental para la obtención de cubrimientos con datos mezclados. In: CIARP 2002, pp. 405–416 (2002)
15. Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In: International Conference on New Methods in Language Processing (1994)
16. Schmid, H.: Improvements in part-of-speech tagging with an application to german. In: ACL SIGDAT-Workshop (1995)
17. Shi, Q., et al.: Semi-Markov Models for Sequence Segmentation. In: Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 640–648 (2007)
18. Stokes, N., Carthy, J., Smeaton, A.: SeLeCT: A Lexical Cohesion Based News Story Segmentation System. AI Communications 17(1), 3–12 (2004)