A Statistical Model for Domain-Independent Text Segmentation

Masao Utiyama and **Hitoshi Isahara**
Communications Research Laboratory
2-2-2 Hikaridai Seika-cho, Soraku-gun,
Kyoto, 619-0289 Japan
mutiyama@crl.go.jp and isahara@crl.go.jp

Abstract

We propose a statistical method that finds the maximum-probability segmentation of a given text. This method does not require training data because it estimates probabilities from the given text. Therefore, it can be applied to any text in any domain. An experiment showed that the method is more accurate than or at least as accurate as a state-of-the-art text segmentation system.

1 Introduction

Documents usually include various topics. Identifying and isolating topics by dividing documents, which is called text segmentation, is important for many natural language processing tasks, including information retrieval (Hearst and Plaunt, 1993; Salton et al., 1996) and summarization (Kan et al., 1998; Nakao, 2000). In information retrieval, users are often interested in particular topics (parts) of retrieved documents, instead of the documents themselves. To meet such needs, documents should be segmented into coherent topics. Summarization is often used for a long document that includes multiple topics. A summary of such a document can be composed of summaries of the component topics. Identification of topics is the task of text segmentation.

A lot of research has been done on text segmentation (Kozima, 1993; Hearst, 1994; Okumura and Honda, 1994; Salton et al., 1996; Yaari, 1997; Kan et al., 1998; Choi, 2000; Nakao, 2000). A major characteristic of the methods used in this research is that they do not require training data to segment given texts. Hearst (1994), for example, used only the similarity of word distributions in a given text to segment the text. Consequently, these methods can be applied to any text in any domain, even if training data do not exist. This property is important when text segmentation is applied to information retrieval or summarization, because both tasks deal with domain-independent documents.

Another application of text segmentation is the segmentation of a continuous broadcast news story into individual stories (Allan et al., 1998). In this application, systems relying on supervised learning (Yamron et al., 1998; Beeferman et al., 1999) achieve good performance because there are plenty of training data in the domain. These systems, however, can not be applied to domains for which no training data exist.

The text segmentation algorithm described in this paper is intended to be applied to the summarization of documents or speeches. Therefore, it should be able to handle domain-independent texts. The algorithm thus does not use any training data. It requires only the given documents for segmentation. It can, however, incorporate training data when they are available, as discussed in Section 5.

The algorithm selects the optimum segmentation in terms of the probability defined by a statistical model. This is a new approach for domain-independent text segmentation. Previous approaches usually used lexical cohesion to segment texts into topics. Kozima (1993), for example, used cohesion based on the spreading activation on a semantic network. Hearst (1994) used the similarity of word distributions as measured by the cosine to gauge cohesion. Reynar (1994) used word repetition as a measure of cohesion. Choi (2000) used the rank of the cosine, rather than the cosine itself, to measure the similarity of sentences.

The statistical model for the algorithm is described in Section 2, and the algorithm for obtaining the maximum-probability segmentation is described in Section 3. Experimental results are presented in Section 4. Further discussion and our conclusions are given in Sections 5 and 6, respectively.

2 Statistical Model for Text Segmentation

We first define the probability of a segmentation of a given text in this section. In the next section, we then describe the algorithm for selecting the most likely segmentation.

Let $W = w_1 w_2 \ldots w_n$ be a text consisting of n words, and let $S = S_1 S_2 \ldots S_m$ be a segmentation of W consisting of m segments. Then the probability of the segmentation S is defined by:

$$\Pr(S|W) = \frac{\Pr(W|S)\Pr(S)}{\Pr(W)}. \tag{1}$$

The most likely segmentation $\hat{S}$ is given by:

$$\hat{S} = \arg\max_{S} \Pr(W|S)\Pr(S), \tag{2}$$

because $\Pr(W)$ is a constant for a given text W.

The definitions of $\Pr(W|S)$ and $\Pr(S)$ are given below, in that order.

2.1 Definition of $\Pr(W|S)$

We define a topic by the distribution of words in that topic. We assume that different topics have different word distributions. We further assume that different topics are statistically independent of each other. We also assume that the words within the scope of a topic are statistically independent of each other given the topic.

Let n_i be the number of words in segment S_i, and let w^i_j be the j-th word in S_i. If we define W_i as

$$W_i = w^i_1 w^i_2 \ldots w^i_{n_i},$$

then $W = W_1 W_2 \ldots W_m$ and $n = \sum_{i=1}^{m} n_i$ hold. This means that S_i and W_i correspond to each other.

Under our assumptions, $\Pr(W|S)$ can be decomposed as follows:

$$\begin{aligned}\Pr(W|S) &= \Pr(W_1 W_2 \ldots W_m|S) \\ &= \prod_{i=1}^{m} \Pr(W_i|S) \\ &= \prod_{i=1}^{m} \Pr(W_i|S_i) \\ &= \prod_{i=1}^{m} \prod_{j=1}^{n_i} \Pr(w^i_j|S_i). \end{aligned} \tag{3}$$

Next, we define $\Pr(w^i_j|S_i)$ as:

$$\Pr(w^i_j|S_i) \equiv \frac{f_i(w^i_j) + 1}{n_i + k}, \tag{4}$$

where $f_i(w^i_j)$ is the number of words in W_i that are the same as w^i_j and k is the number of different words in W. For example, if $W = W_1 W_2$, where $W_1 = ababa$ and $W_2 = cccdcc$, then $f_1(a) = 3$, $f_1(b) = 2$, $f_2(c) = 5$, $f_2(d) = 1$, and $k = 4$. Equation (4) is known as Laplace's law (Manning and Schütze, 1999).

$f_i(w^i_j)$ can be defined as:

$$f_i(w^i_j) \equiv h(w^i_j|w^i_1 w^i_2 \ldots w^i_{n_i}) \tag{5}$$

for

$$h(w^i_j|w^i_1 w^i_2 \ldots w^i_{n_i}) \equiv \sum_{k=1}^{n_i} \delta(w^i_k, w^i_j), \tag{6}$$

where $\delta(w^i_k, w^i_j) = 1$ when w^i_k and w^i_j are the same word and $\delta(w^i_k, w^i_j) = 0$ otherwise. For example, $h(a|baba) = \delta(b,a) + \delta(a,a) + \delta(b,a) + \delta(a,a) = 0 + 1 + 0 + 1 = 2$.

Equations (5) and (6) are used in Section 3 to describe the algorithm for finding the maximum-probability segmentation.

2.2 Definition of $\Pr(S)$

The definition of $\Pr(S)$ can vary depending on our prior information about the possibility of segmentation S. For example, we might know the average length of the segments and want to incorporate it into $\Pr(S)$.

Our assumption, however, is that we do not have such prior information. Thus, we have to use some uninformative prior probability.

We define $\Pr(S)$ as

$$\Pr(S) \equiv n^{-m} \tag{7}$$

Equation (7) is determined on the basis of its *description length*,[1] $l(S)$; i.e.,

$$\Pr(S) = 2^{-l(S)} \tag{8}$$

where $l(S) = m \log n$ bits.[2] This description length is derived as follows:

> Suppose that there are two people, a sender and a receiver, both of whom know the text to be segmented. Only the sender knows the exact segmentation, and he/she should send a message so that the receiver can segment the text correctly. To this end, it is sufficient for the sender to send m integers, i.e., $n_1, n_2, \ldots, n_m$, because these integers represent the lengths of segments and thus uniquely determine the segmentation once the text is known.
>
> A segment length n_i can be encoded using $\log n$ bits, because n_i is a number between 1 and n. The total description length for all the segment lengths is thus $m \log n$ bits.[3]

Generally speaking, $\Pr(S)$ takes a large value when the number of segments is small. On the other hand, $\Pr(W|S)$ takes a large value when the number of segments is large. If only $\Pr(W|S)$ is used to segment the text, then the resulting segmentation will have too many segments. By using both $\Pr(S)$ and $\Pr(W|S)$, we can get a reasonable number of segments.

3 Algorithm for Finding the Maximum-Probability Segmentation

To find the maximum-probability segmentation $\hat{S}$, we first define the cost of segmentation S as

$$C(S) \equiv -\log \Pr(W|S)\Pr(S), \tag{9}$$

and we then minimize $C(S)$ to obtain $\hat{S}$, because

$$\hat{S} = \arg\max_S \Pr(W|S)\Pr(S) = \arg\min_S C(S). \tag{10}$$

$C(S)$ can be decomposed as follows:

$$\begin{aligned} C(S) &= -\log \Pr(W|S)\Pr(S) \\ &= -\sum_{i=1}^{m}\sum_{j=1}^{n_i} \log \Pr(w_j^i|S_i) - \log \Pr(S) \\ &= -\sum_{i=1}^{m}\sum_{j=1}^{n_i} \log \frac{f_i(w_j^i)+1}{n_i+k} + m \log n \\ &= \sum_{i=1}^{m} c(w_1^i w_2^i \ldots w_{n_i}^i | n, k), \end{aligned} \tag{11}$$

where

$$\begin{aligned} & c(w_1^i w_2^i \ldots w_{n_i}^i | n, k) \\ & \quad \equiv \sum_{j=1}^{n_i} \log \frac{n_i+k}{f_i(w_j^i)+1} + \log n. \end{aligned} \tag{12}$$

We further rewrite Equation (12) in the form of Equation (13) below by using Equation (5) and replacing n_i with $\#(w_1^i w_2^i \ldots w_{n_i}^i)$, where $\#(words)$ is the length of *words*, i.e.,the number of word tokens in *words*. Equation (13) is used to describe our algorithm in Section 3.1:

$$\begin{aligned} & c(w_1^i w_2^i \ldots w_{n_i}^i | n, k) \\ & \quad = \sum_{j=1}^{\#(w_1^i w_2^i \ldots w_{n_i}^i)} \log \frac{\#(w_1^i w_2^i \ldots w_{n_i}^i) + k}{h(w_j^i | w_1^i w_2^i \ldots w_{n_i}^i) + 1} \\ & \quad + \log n. \end{aligned} \tag{13}$$

3.1 Algorithm

This section describes an algorithm for finding the minimum-cost segmentation. First, we define the terms and symbols used to describe the algorithm.

Given a text $W = w_1 w_2 \ldots w_n$ consisting of n words, we define g_i as the position between w_i and w_{i+1}, so that g_0 is just before w_1 and g_n is just after w_n.

Next, we define a graph $G = \langle V, E \rangle$, where V is a set of nodes and E is a set of edges. V is defined as

$$V = \{g_i | 0 \le i \le n\} \tag{14}$$

[1] Stolcke and Omohundro uses description length priors to induce the structure of hidden Markov models (Stolcke and Omohundro, 1994).

[2] 'Log' denotes the logarithm to the base 2.

[3] We have used $\frac{m}{2}\log n$ as $l(S)$ before. But we use $m \log n$ in this paper, because it is easily interpreted as a description length and the experimental results obtained by using $m \log n$ are slightly better than those obtained by using $\frac{m}{2}\log n$. An anonymous reviewer suggests using a Poisson distribution whose parameter is $\frac{n}{m}$, the average length of a segment (in words), as prior probability. We leave it for future work to compare the suitability of various prior probabilities for text segmentation.

and E is defined as

$$E = \{e_{ij} | 0 \leq i < j \leq n\}, \quad (15)$$

where the edges are ordered; the initial vertex and the terminal vertex of e_{ij} are g_i and g_j, respectively. An example of G is shown in Figure 1.

We say that e_{ij} covers $w_{i+1}w_{i+2}\ldots w_j$. This means that e_{ij} represents a segment $w_{i+1}w_{i+2}\ldots w_j$. Thus, we define the cost c_{ij} of edge e_{ij} by using Equation (13):

$$c_{ij} = c(w_{i+1}w_{i+2}\ldots w_j | n, k), \quad (16)$$

where k is the number of different words in W.

Given these definitions, we describe the algorithm to find the minimum-cost segmentation or maximum-probability segmentation as follows:

Step 1. Calculate the cost c_{ij} of edge e_{ij} for $0 \leq i < j \leq n$ by using Equation (16).

Step 2. Find the minimum-cost path from g_0 to g_n.

Algorithms for finding the minimum-cost path in a graph are well known. An algorithm that can provide a solution for Step 2 will be a simpler version of the algorithm used to find the maximum-probability solution in Japanese morphological analysis (Nagata, 1994). Therefore, a solution can be obtained by applying a dynamic programming (DP) algorithm.[4] DP algorithms have also been used for text segmentation by other researchers (Ponte and Croft, 1997; Heinonen, 1998).

The path thus obtained represents the minimum-cost segmentation in G when edges correspond with segments. In Figure 1, for example, if $e_{01}e_{13}e_{35}$ is the minimum-cost path, then $[w_1][w_2w_3][w_4w_5]$ is the minimum-cost segmentation.

The algorithm automatically determines the number of segments. But the number of segments can also be specified explicitly by specifying the number of edges in the minimum-cost path.

The algorithm allows the text to be segmented anywhere between words; i.e., all the positions between words are candidates for segment boundaries. It is easy, however, to modify the algorithm so that the text can only be segmented at particular positions, such as the ends of sentences or paragraphs. This is done by using a subset of E in Equation (15). We use only the edges whose initial and terminal vertices are candidate boundaries that meet particular conditions, such as being the ends of sentences or paragraphs. We then obtain the minimum-cost path by doing Steps 1 and 2. The minimum-cost segmentation thus obtained meets the boundary conditions. In this paper, we assume that the segment boundaries are at the ends of sentences.

3.2 Properties of the segmentation

Generally speaking, the number of segments obtained by our algorithm is not sensitive to the length of a given text, which is counted in words. In other words, the number of segments is relatively stable with respect to variation in the text length. For example, the algorithm divides a newspaper editorial consisting of about 27 sentences into 4 to 6 segments, while on the other hand, it divides a long text consisting of over 1000 sentences into 10 to 20 segments. Thus, the number of segments is not proportional to text length. This is due to the term $m \log n$ in Equation (11). The value of this term increases as the number of words increases. The term thus suppresses the division of a text when the length of the text is long.

This stability is desirable for summarization, because summarizing a given text requires selecting a relatively small number of topics from it. If a text segmentation system divides a given text into a relatively small number of segments, then a summary of the original text can be composed by combining summaries of the component segments (Kan et al., 1998; Nakao, 2000). A finer segmentation can be obtained by applying our algorithm recursively to each segment, if necessary.[5]

[4] A program that implements the algorithm described in this section is available at `http://www.crl.go.jp/jt/a132/members/mutiyama/softwares.html`

[5] We segmented various texts without rigorous evaluation and found that our method is good at segmenting a text into a relatively small number of segments. On the other hand, the method is not good at segmenting a text into a large number of segments. For example, the method is good at segmenting a 1000-sentence text into 10 segments. In such a case, the segment boundaries seem to correspond well with topic boundaries. But, if the method is forced to segment the same text into 50 segments by specifying the number of

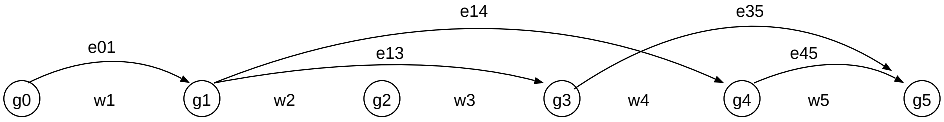

Figure 1: Example of a graph.

4 Experiments

4.1 Material

We used publicly available data to evaluate our system. This data was used by Choi (2000) to compare various domain-independent text segmentation systems.[6] He evaluated $C99$ (Choi, 2000), TextTiling (Hearst, 1994), DotPlot (Reynar, 1998), and Segmenter (Kan et al., 1998) by using the data and reported that $C99$ achieved the best performance among these systems.

The data description is as follows: "An artificial test corpus of 700 samples is used to assess the accuracy and speed performance of segmentation algorithms. A sample is a concatenation of ten text segments. A segment is the first n sentences of a randomly selected document from the Brown corpus. A sample is characterised by the range n." (Choi, 2000) Table 1 gives the corpus statistics.

Range of n	$3-11$	$3-5$	$6-8$	$9-11$
# samples	400	100	100	100

Table 1: Test corpus statistics. (Choi, 2000)

Segmentation accuracy was measured by the probabilistic error metric P_k proposed by Beeferman, et al. (1999).[7] Low P_k indicates high accuracy.

4.2 Experimental procedure and results

The sample texts were preprocessed – i.e., punctuation and stop words were removed and the remaining words were stemmed – by a program using the libraries available in Choi's package. The texts were then segmented by the systems listed in Tables 2 and 3. The segmentation boundaries were placed at the ends of sentences. The segmentations were evaluated by applying an evaluation program in Choi's package.

The results are listed in Tables 2 and 3. $U00$ is the result for our system when the numbers of segments were determined by the system. $U00_{(b)}$ is the result for our system when the numbers of segments were given beforehand.[8] $C99$ and $C99_{(b)}$ are the corresponding results for the systems described in Choi's paper (Choi, 2000).[9]

	$3-11$	$3-5$	$6-8$	$9-11$	Total
$U00$	11%**	13%**	6%**	6%**	10%**
$C99$	13%	18%	10%	10%	13%
prob	7.9E-5	4.9E-3	2.5E-5	7.5E-8	9.7E-12

Table 2: Comparison of P_k: the numbers of segments were determined by the systems.

In these tables, the symbol "$**$" indicates that the difference in P_k between the two systems is statistically significant at the 1% level, based on

edges in the minimum-cost path, then the resulting segmentation often contains very small segments consisting of only one or two sentences. We found empirically that segments obtained by recursive segmentation were better than those obtained by minimum-cost segmentation when the specified number of segments was somewhat larger than that of the minimum-cost path, whose number of segments was automatically determined by the algorithm.

[6] The data is available from
`http://www.cs.man.ac.uk/˜choif/software/C99-1.2-release.tgz.`
We used
`naacl00Exp/data/{1,2,3}/{3-11,3-5,6-8,9-11}/*`,
which is contained in the package, for our experiment.

[7] Let `ref` be a correct segmentation and let `hypo` be a segmentation proposed by a text segmentation system: Then the "number $P_k(\texttt{ref}, \texttt{hypo})$ is the probability that a randomly chosen pair of words a distance of k words apart is inconsistently classified; that is, for one of the segmentations the pair lies in the same segment, while for the other the pair spans a segment boundary" (Beeferman et al., 1999), where k is chosen to be half the average reference segment length (in words).

[8] If two segmentations have the same cost, then our systems arbitrarily select one of them; i.e., the systems select the segmentation processed previously.

[9] The results for $C99_{(b)}$ in Table 3 are slightly different from those listed in Table 6 of Choi's paper (Choi, 2000). This is because the original results in that paper were based on 500 samples, while the results in our Table 3 were based on 700 samples (Choi, personal communication).

	$3-11$	$3-5$	$6-8$	$9-11$	Total
$U00_{(b)}$	10%**	9%	7%**	5%**	9%**
$C99_{(b)}$	12%	11%	10%	9%	11%
prob	2.7E-4	0.080	2.3E-3	1.0E-4	6.8E-9

Table 3: Comparision of P_k: the numbers of segments were given beforehand.

a one-sided t-test of the null hypothesis of equal means. The probability of the null hypothesis being true is displayed in the row indicated by "prob". The column labels, such as "$3-5$", indicate that the numbers in the column are the averages of P_k over the corresponding sample texts. "Total" indicates the averages of P_k over all the text samples.

These tables show statistically that our system is more accurate than or at least as accurate as $C99$. This means that our system is more accurate than or at least as accurate as previous domain-independent text segmentation systems, because $C99$ has been shown to be more accurate than previous domain-independent text segmentation systems.[10]

5 Discussion

5.1 Evaluation

Evaluation of the output of text segmentation systems is difficult because the required segmentations depend on the application. In this paper, we have used an artificial corpus to evaluate our system. We regard this as appropriate for comparing relative performance among systems.

It is important, however, to assess the performance of systems by using real texts. These texts should be domain independent. They should also be multi-lingual if we want to test the multilinguality of systems. For English, Klavans, et al. describe a segmentation corpus in which the texts were segmented by humans (Klavans et al., 1998). But, there are no such corpora for other languages. We are planning to build a segmentation corpus for Japanese, based on a corpus of speech transcriptions (Maekawa and Koiso, 2000).

5.2 Related work

Our proposed algorithm finds the maximum-probability segmentation of a given text. This is a new approach for domain-independent text segmentation. A probabilistic approach, however, has already been proposed by Yamron, et al. for domain-dependent text segmentation (broadcast news story segmentation) (Yamron et al., 1998). They trained a hidden Markov model (HMM), whose states correspond to topics. Given a word sequence, their system assigns each word a topic so that the maximum-probability topic sequence is obtained. Their model is basically the same as that used for HMM part-of-speech (POS) taggers (Manning and Schütze, 1999), if we regard topics as POS tags.[11] Finding topic boundaries is equivalent to finding topic transitions; i.e., a continuous topic or segment is a sequence of words with the same topic.

Their approach is indirect compared with our approach, which directly finds the maximum-probability segmentation. As a result, their model can not straightforwardly incorporate features pertaining to a segment itself, such as the average length of segments. Our model, on the other hand, can incorporate this information quite naturally. Suppose that the length of a segment x follows a normal distribution $N(x;\mu,\sigma)$, with a mean of μ and standard deviation of σ (Ponte and Croft, 1997). Then Equation (13) can be augmented to

$$\begin{aligned}
&c(w^i_1 w^i_2 \ldots w^i_{n_i} | n, k, \mu, \sigma, \alpha, \beta, \gamma) \\
&\quad = \alpha \sum_{j=1}^{\#(w^i_1 w^i_2 \ldots w^i_{n_i})} \log \frac{\#(w^i_1 w^i_2 \ldots w^i_{n_i}) + k}{h(w^i_j | w^i_1 w^i_2 \ldots w^i_{n_i}) + 1} \\
&\qquad + \beta \log n \\
&\qquad + \gamma \log \frac{1}{N(\#(w^i_1 w^i_2 \ldots w^i_{n_i}) | \mu, \sigma)}, \qquad (17)
\end{aligned}$$

[10] Speed performance is not our main concern in this paper. Our implementations of $U00$ and $U00b$ are not optimum. However, $U00$ and $U00b$, which are implemented in C, run as fast as $C99$ and $C99b$, which are implemented in Java (Choi, 2000), due to the difference in programming languages. The average run times for a sample text were

$$\begin{aligned}
U00 &= 1.36 \text{ sec.} \\
C99 &= 1.49 \text{ sec.} \\
U00b &= 1.37 \text{ sec.} \\
C99b &= 1.45 \text{ sec.}
\end{aligned}$$

on a Pentium III 750-MHz PC with 384-MB RAM running RedHat Linux 6.2.

[11] The details are different, though.

where $\alpha + \beta + \gamma = 1$. Equation (17) favors segments whose lengths are similar to the average length (in words).

Another major difference from their algorithm is that our algorithm does not require training data to estimate probabilities, while their algorithm does. Therefore, our algorithm can be applied to domain-independent texts, while their algorithm is restricted to domains for which training data are available. It would be interesting, however, to compare our algorithm with their algorithm for the case when training data are available. In such a case, our model should be extended to incorporate various features such as the average segment length, clue words, named entities, and so on (Reynar, 1999; Beeferman et al., 1999).

Our proposed algorithm naturally estimates the probabilities of words in segments. These probabilities, which are called word densities, have been used to detect important descriptions of words in texts (Kurohashi et al., 1997). This method is based on the assumption that the density of a word is high in a segment in which the word is discussed (defined and/or explained) in some depth. It would be interesting to apply our method to this application.

6 Conclusion

We have proposed a statistical model for domain-independent text segmentation. This method finds the maximum-probability segmentation of a given text. The method has been shown to be more accurate than or at least as accurate as previous methods. We are planning to build a segmentation corpus for Japanese and evaluate our method against this corpus.

Acknowledgements

We thank Freddy Y. Y. Choi for his text segmentation package.

References

James Allan, Jaime Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. 1998. Topic detection and tracking pilot study final report. In *Proc. of the DARPA Broadcast News Transcription and Understanding Workshop*.

Doug Beeferman, Adam Berger, and John Lafferty. 1999. Statistical models for text segmentation. *Machine Learning*, 34(1-3):177–210.

Freddy Y. Y. Choi. 2000. Advances in domain independent linear text segmentation. In *Proc. of NAACL-2000*.

Marti A. Hearst and Christian Plaunt. 1993. Subtopic structuring for full-length document access. In *Proc. of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 59–68.

Marti A. Hearst. 1994. Multi-paragraph segmentation of expository text. In *Proc. of ACL'94*.

Oskari Heinonen. 1998. Optimal multi-paragraph text segmentation by dynamic programming. In *Proc. of COLING-ACL'98*.

Min-Yen Kan, Judith L. Klavans, and Kathleen R. McKeown. 1998. Linear segmentation and segment significance. In *Proc. of WVLC-6*, pages 197–205.

Judith L. Klavans, Kathleen R. McKeown, Min-Yen Kan, and Susan Lee. 1998. Resources for the evaluation of summarization techniques. In *Proceedings of the 1st International Conference on Language Resources and Evaluation (LREC)*, pages 899–902.

Hideki Kozima. 1993. Text segmentation based on similarity between words. In *Proc. of ACL'93*.

Sadao Kurohashi, Nobuyuki Shiraki, and Makoto Nagao. 1997. A method for detecting important descriptions of a word based on its density distribution in text (in Japanese). *IPSJ (Information Processing Society of Japan) Journal*, 38(4):845–854.

Kikuo Maekawa and Hanae Koiso. 2000. Design of spontaneous speech corpus for Japanese. In *Proc of International Symposium: Toward the Realization of Spontaneous Speech Engineering*, pages 70–77.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press.

Masaaki Nagata. 1994. A stochastic Japanese morphological analyzer using a forward-DP backward-A* n-best search algorithm. In *Proc. of COLING'94*, pages 201–207.

Yoshio Nakao. 2000. An algorithm for one-page summarization of a long text based on thematic hierarchy detection. In *Proc. of ACL'2000*, pages 302–309.

Manabu Okumura and Takeo Honda. 1994. Word sense disambiguation and text segmentation based on lexical cohesion. In *Proc. of COLING-94*.

Jay M. Ponte and W. Bruce Croft. 1997. Text segmentation by topic. In *Proc. of the First European Conference on Research and Advanced Technology for Digital Libraries*, pages 120–129.

Jeffrey C. Reynar. 1994. An automatic method of finding topic boundaries. In *Proc. of ACL-94*.

Jeffrey C. Reynar. 1998. *Topic segmentation: Algorithms and applications*. Ph.D. thesis, Computer and Information Science, University of Pennsylvania.

Jeffrey C. Reynar. 1999. Statistical models for topic segmentation. In *Proc. of ACL-99*, pages 357–364.

Gerard Salton, Amit Singhal, Chris Buckley, and Mandar Mitra. 1996. Automatic text decomposition using text segments and text themes. In *Proc. of Hypertext'96*.

Andreas Stolcke and Stephen M. Omohundro. 1994. Best-first model merging for hidden Markov model induction. Technical Report TR-94-003, ICSI, Berkeley, CA.

Yaakov Yaari. 1997. Segmentation of expository texts by hierarchical agglomerative clustering. In *Proc. of the Recent Advances in Natural Language Processing*.

J. P. Yamron, I. Carp, S. Lowe, and P. van Mulbregt. 1998. A hidden Markov model approach to text segmentation and event tracking. In *Proc. of ICASSP-98*.