A Critique and Improvement of an Evaluation Metric for Text Segmentation

Lev Pevzner*
Harvard University

Marti Hearst†
UC Berkeley

The P_k evaluation metric, initially proposed by Beeferman et al. 1997, is becoming the standard measure for assessing text segmentation algorithms. However, a theoretical analysis of the metric finds several problems: the metric penalizes false negatives more heavily than false positives, over-penalizes near-misses, and is affected by variation in segment size distribution. We propose a simple modification to the P_k metric that remedies these problems. This new metric – called WindowDiff – moves a fixed-sized window across the text, and penalizes the algorithm whenever the number of boundaries within the window does not match the true number of boundaries for that window of text.

1. Introduction

Text segmentation is the task of determining the positions at which topics change in a stream of text. Interest in automatic text segmentation has blossomed over the last few years, with applications ranging from information retrieval to text summarization to story segmentation of video feeds. Early work in multi-paragraph discourse segmentation examined the problem of subdividing texts into multi-paragraph units that represent passages or subtopics. An example, drawn from (Hearst, 1997) is a 21-paragraph science news article, called *Stargazers*, whose main topic is the existence of life on earth and other planets. Its contents can be described as consisting of the following subtopic discussions (numbers indicate paragraphs):

1–3	*Intro – the search for life in space*
4–5	*The moon's chemical composition*
6–8	*How early earth-moon proximity shaped the moon*
9–12	*How the moon helped life evolve on earth*
13	*Improbability of the earth-moon system*
14–16	*Binary/trinary star systems make life unlikely*
17–18	*The low probability of non-binary/trinary systems*
19–20	*Properties of earth's sun that facilitate life*
21	*Summary*

* 380 Leverett Mail Center, Cambridge, MA 02138
† 102 South Hall #4600, Berkeley, CA 94720

The TextTiling algorithm (Hearst, 1993; Hearst, 1994; Hearst, 1997) attempts to recognize these subtopic changes by making use of patterns of lexical co-occurrence and distribution; subtopic boundaries are assumed to occur at the point in the documents at which large shifts in vocabulary occur. Many others have used this technique, or slight variations of it for subtopic segmentation (Nomoto and Nitta, 1994; Hasnah, 1996; Richmond, Smith, and Amitay, 1997; Heinonen, 1998; Boguraev and Neff, 2000). Other techniques use clustering and/or similarity matrices based on word cooccurrences (Reynar, 1994; Choi, 2000; Yaari, 1997), while still others use machine-learning techniques to detect cue words, or hand-selected cue words to detect segment boundaries (Passonneau and Litman, 1993; Beeferman, Berger, and Lafferty, 1997; Manning, 1998).

Researchers have explored the use of this kind of document segmentation to improve automated summarization (Salton et al., 1994; Barzilay and Elhadad, 1997; Kan, Klavans, and Mckeown, 1998; Mittal et al., 1999; Boguraev and Neff, 2000) and automated genre detection (Karlgren, 1996). Text segmentation issues are also important for passage retrieval, a subproblem of information retrieval (Hearst and Plaunt, 1993; Salton, Allan, and Buckley, 1993; Callan, 1994; Kaszkiel and Zobel, 1997). More recently, a great deal of interest has arisen in using automatic segmentation for the detection of topic and story boundaries in news feeds (Mani et al., 1997; Merlion, Morey, and Maybury, 1997; Ponte and Croft, 1997; Hauptmann and Whitbrock, 1998; Allan et al., 1998; Beeferman, Berger, and Lafferty, 1997; Beeferman, Berger, and Lafferty, 1999). Sometimes segmentation is done at the clause level, for the purposes of detecting nuances of dialogue structure or for more sophisticated discourse processing purposes (Morris and Hirst, 1991; Passonneau and Litman, 1993; Litman and Passonneau, 1995; Hirschberg and Nakatani, 1996; Marcu, 2000). Some of these algorithms produce hierarchical dialogue segmentations whose evaluation is outside the scope of this discussion.

1.1 Evaluating Segmentation Algorithms

There are two major difficulties associated with evaluating algorithms for text segmentation. The first is that since human judges do not always agree where boundaries should be placed and how fine-grained an analysis should be, it is difficult to choose a reference segmentation for comparison. Some evaluations circumvent this difficulty by detecting boundaries in sets of concatenated documents, where there can be no disagreements about the fact of the matter (Reynar, 1994; Choi, 2000); others have several human judges make ratings to produce a ”gold standard”.

The second difficulty with evaluating these algorithms is that for different applications of text segmentation, different kinds of errors become important. For instance, for information retrieval, it can be acceptable for boundaries to be off by a few sentences – a condition called a near-miss – but for news boundary detection, accurate placement is crucial. For this reasons, some researchers prefer not to measure the segmentation algorithm directly, but consider its impact on the end application (Manning, 1998; Kan, Klavans, and Mckeown, 1998). Our approach to these two difficulties is to evaluate algorithms on real segmentations using a ”gold standard”, and to develop an evaluation algorithm which suits all applications reasonably well.

Precision and recall are standard evaluation measures for information retrieval tasks, and are often applied to evaluation of text segmentation algorithms as well. Precision is the percentage of boundaries identified by an algorithm that are indeed true boundaries, while recall is the percentage of true boundaries that are identified by the algorithm. However, precision and recall are problematic for two reasons. The first is that there is an inherent tradeoff between precision and recall; improving one tends to cause the score for the other to decline. In the segmentation example, positing more boundaries will tend to improve the recall while at the same time reducing the precision. Some evaluators use a weighted combination of the two known as the F-measure (Baeza-Yates and Ribeiro-Neto, 1999), but this is difficult to interpret (Beeferman, Berger, and Lafferty, 1999). Another approach is to plot a precision-recall curve, showing the scores for precision at different levels of recall.

Another problem with precision and recall is that they are not sensitive to near-misses. Consider, for example, a reference segmentation and the results obtained by two different text segmentation algorithms, as depicted in Figure 1. In both cases the algorithms fail to match any boundary precisely; both receive scores of 0 for precision and recall. However, Algorithm A-0 is close to correct in almost all cases, whereas Algorithm A-1 is entirely off, adding extraneous boundaries and missing important boundaries entirely. In some circumstances it would be useful to have an evaluation metric that penalizes A-0 less harshly than A-1.

1.2 The P_k Evaluation Metric

Beeferman, Berger, and Lafferty (1997) introduce a new evaluation metric that attempts to resolve the problems with precision and recall, including assigning partial credit to near misses. They justify their metric as follows:

> Segmentation ... is about identifying boundaries between successive units of information in a text corpus. Two such units are either related or unrelated by the intent of the document author. A natural way to reason about developing a segmentation algorithm is therefore to optimize the likelihood that two such units are correctly labeled as being related or being unrelated. Our error metric P_μ is simply the *probability that two sentences drawn randomly from the corpus are correctly identified as belonging to the same document or not belonging to the same document*.

The derivation of P_μ is rather involved, and a much simpler version is adopted in the later work (Beeferman, Berger, and Lafferty, 1999) and by others. This version

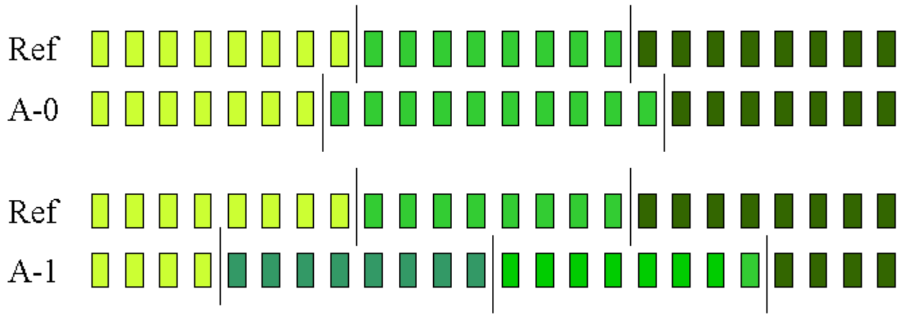

Figure 1
Two hypothetical segmentations of the same reference (ground truth) document segmentation. The boxes indicate sentences or other units of subdivision, and spaces between boxes indicate potential boundary locations. Algorithm A-0 makes two near-misses, while Algorithm A-1 misses both boundaries by a wide margin and introduces three false positives. Both algorithms would receive scores of 0 for both precision and recall.

is referred to as P_k, and is calculated by setting k to half of the average true segment size, and then computing penalties via a moving window of length k. At each location, the algorithm determines whether the two ends of the probe are in the same or different segments in the reference segmentation, and increases a counter if the algorithm's segmentation disagrees. The resulting count is scaled between 0 and 1 by dividing by the number of measurements taken. An algorithm that assigns all boundaries correctly receives a score of 0. Beeferman, Berger, and Lafferty (1999) state as part of the justification for this metric that, to discourage "cheating" of the metric, degenerate algorithms – those that places boundaries at every position, or places no boundaries at all – are assigned (approximately) the same score. Additionally, the authors define a *false negative* (also referred to as a *miss*) as a case when a boundary is present in the reference segmentation but missing in the algorithm's hypothesized segmentation, and a *false positive* as an assignment of a boundary that does not exist in the reference segmentation.

2. Analysis of the P_k Error Metric

The P_k metric is fast becoming the standard among researchers working in text segmentation (Allan et al., 1998; Dharanipragada et al., 1999; Eichmann et al., 1999; van Mulbregt et al., 1999; Choi, 2000). However, we have reservations about this metric. We claim that the fundamental premise behind it is flawed, and additionally, it has several significant drawbacks, which we identify in this section. In the remainder of the paper we suggest modifications to resolve these problems, and report the results of simulations that validate the analysis and suggest that the modified metric is an improvement

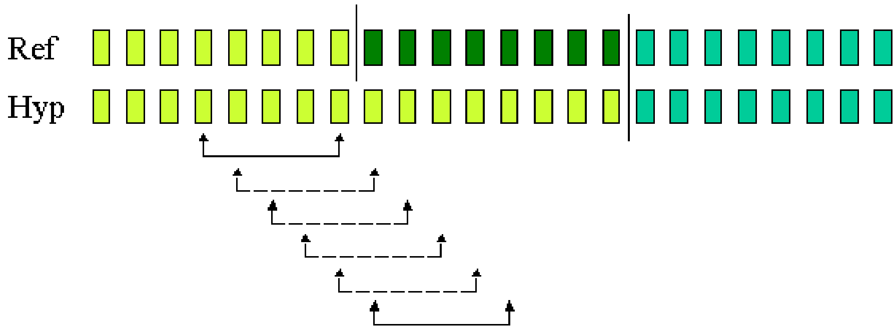

Figure 2
An illustration of how the P_k metric handles false negatives. The arrowed lines indicate the two poles of the probe as it moves from left to right, the boxes indicate sentences or other units of subdivision, and the width of the window (k) is four, meaning four potential boundaries fall between the two ends of the probe. Solid lines indicate no penalty is assigned, dashed lines indicate a penalty is assigned. Total penalty is always k for false negatives.

over the original.

Problem 1 - False negatives penalized more than false positives
Assume a text with segments of average size $2k$, where k is the distance between the two ends of the P_k probe. If the algorithm misses a boundary – produces a false negative – it receives k penalties. To see why, suppose S1 and S2 are two segments of length $2k$, and the algorithm misses the transition from S1 to S2. When P_k sweeps across S1, if both ends of the probe point to sentences that are inside S1, the two sentences are in the same segment in both the reference and the hypothesis, and no penalty is incurred. When the right end of the probe crosses the reference boundary between S1 and S2, it will start recording non-matches, since the algorithm assigns the two sentences to the same segment, while the reference does not. This circumstance happens k times, until both ends of the probe point to sentences that are inside S2. (See Figure 2.) This analysis assumes average size segments; variation in segment size is discussed below, but does not have a large effect on this result.

Now, consider false positives. A false positive occurs when the algorithm places a boundary at some position where there is no boundary in the reference segmentation. The number of times that this false positive is noted by P_k is dependent on where exactly inside S2 the false positive occurs. (See Figure 3.) If it occurs in the middle of the segment, the false positive is noted k times (as seen on the righthand side of Figure 3). If it occurs $j < k$ sentences from the beginning or the end of the segment, it is penalized j times. Assuming uniformly distributed false positives, on average a false positive is

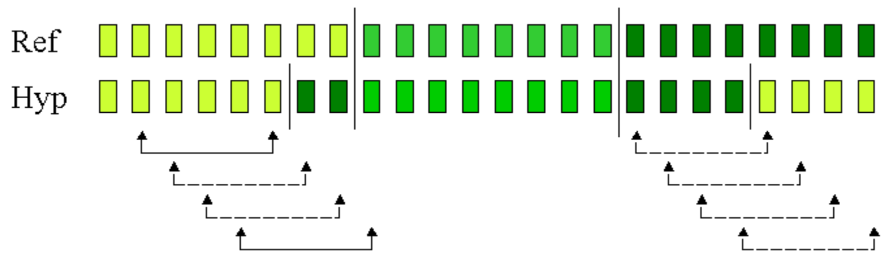

Figure 3
An illustration of how the P_k metric handles false positives. Notation is as in Figure 2. Total penalty depends on the distance between the false positive and the relevant correct boundaries; on average it is $k/2$ assuming a uniform distribution of boundaries across the document. This example shows the consequences of two different locations of false positives; on the left the penalty is $k/2$, on the right it is k.

noted $\frac{k}{2}$ times by the metric – half the rate for false negatives. This average increases with segment size, as will be discussed later, and changes if we assume different distributions of false positives throughout the document. However, this does not change the fact that in most cases false positives are penalized some amount less than false negatives.

This is not an entirely undesirable side effect. This metric was devised to take into account how close an assigned boundary is to the true one, rather than just marking it as correct or incorrect. This method of penalizing false positives achieves this goal – the closer the algorithm's boundary is to the actual boundary, the less it is penalized. However, over-penalizing false negatives to do this is not desirable.

One way to fix the problem of penalizing false negatives more than false positives is to double the false positive penalty (or halve the false negative penalty). However this would undermine the probabilistic nature of the metric. In addition, doubling the penalty may not always be the correct solution, since segment size will vary from the average, and false positives are not necessarily uniformly distributed throughout the document.

Problem 2 - Number of boundaries ignored

Another important problem with the P_k metric is that it allows some errors to go unpenalized. In particular, it does not take into account the number of segment boundaries between the two ends of the probe. (See Figure 4.) Let r_i indicate the number of boundaries between the ends of the probe according to the reference segmentation, and let a_i indicate the number of boundaries proposed by some text segmentation algorithm for the same stretch of text. If $r_i = 1$ (the reference segmentation indicates one boundary)

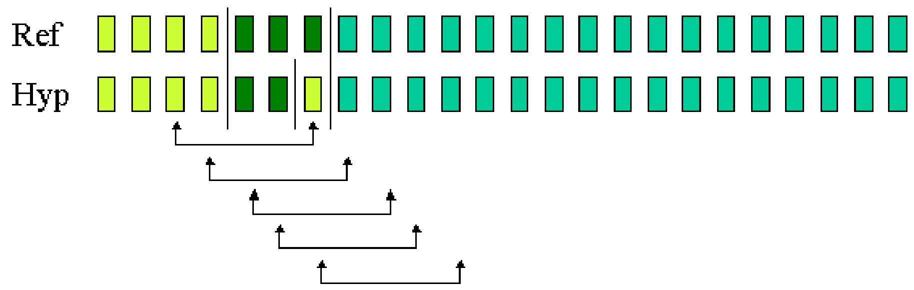

Figure 4
An illustration of the fact that the P_k metric fails to penalize false positives that fall within k sentences of a true boundary. Notation is as in Figure 2.

and $a_i = 2$ (the algorithm marks two boundaries within this range) then the algorithm makes at least one false positive (spurious boundary) error. However, the evaluation metric P_k does not assign a penalty in this situation. Similarly, if $r_i = 2$ and $a_i = 1$, the algorithm has made at least one false negative (missing boundary) error, but is not penalized for this error under P_k.

Problem 3 - Sensitivity to variations in segment size
The size of the segment plays a role in the amount that a false positive within the segment or a false negative at its boundary is penalized. Let us consider false negatives (missing boundaries) first. As seen above, with average size segments, the penalty for a false negative is k. For larger segments, it remains at k – it cannot be any larger than that, since for a given position i there can be at most k intervals of length k that include that position. As segment size gets smaller, however, the false negative penalty changes. Suppose we have two segments, A and B, and the algorithm misses the boundary between them. Then the algorithm will be penalized k times if $Size(A) + Size(B) > 2k$, i.e., as long as each segment is about half the average size or larger. The penalty will then decrease linearly with $Size(A) + Size(B)$ so long as $k < Size(A) + Size(B) < 2k$. To be more exact, the penalty actually decreases linearly as the size of either segment decreases below k. This is intuitively clear from the simple observation that in order to incur a penalty at any range r_i for a false negative, it has to be the case that $r_i > a_i$. In order for this to be true, both the segment to the left and to the right of the missed boundary has to be of size greater than k, or else the penalty can only be equal to the size of the smaller segment. When $Size(A) + Size(B) < k$, the penalty disappears completely, since then the probe's interval is larger than the combined size of both segments, making it not sensitive enough to detect the false negative. It should be noted that fixing Problem 2 would at least partially fix this bias as well.

Now, consider false positives (extraneous boundaries). For average segment size and a uniform distribution of false positives, the average penalty is $\frac{k}{2}$, as described earlier. In general, in large enough segments, the penalty when the false positive is a distance $d < k$ from a boundary is d, and the penalty when the false positive is a distance $d > k$ from a boundary is k. Thus, for larger segments, the average penalty assuming a uniform distribution becomes larger, because there are more places in the segment that are at least k positions away from a boundary. The behavior at the edges of the segments remains the same, though, so average penalty never reaches k. Now consider what happens with smaller segments. Suppose we have a false positive in segment A. As $Size(A)$ decreases from $2k$ to k, the average false positive penalty decreases linearly with it, because when $Size(A)$ decreases below $2k$, the maximum distance any sentence can be from a boundary becomes less than k. Therefore, the maximum possible penalty for a false positive in A is less than k, and this number continues to decrease as $Size(A)$ decreases. When $Size(A) < k$, the false positive penalty disappears, for the same reason as the false negative penalty disappears for smaller segments. Again, fixing Problem 2 would go a long way toward eliminating this bias.

Thus errors in larger-than-average segments increase the penalty slightly (for false positives) or not at all (for false negatives) as compared to average size segments, while errors in smaller-than-average segments decrease the penalty significantly for both types of error. This means that as the variation of segment size increases, the metric becomes more lenient, since it severely under-penalizes errors in smaller segments, while not making up for this by over-penalizing errors in larger segments.

Problem 4 - Near-miss error penalized too much

Reconsider the segmentation made by algorithm A-0 in Figure 1. In both cases of boundary assignment, algorithm A-0 makes both a false positive and a false negative error, but places the boundary very close to the actual one. We will call this kind of error a *near-miss error*, distinct from a false positive or false negative error. Distinguishing this type of error from "pure" false positives better reflects the goal of creating a metric different from precision and recall, since it can be penalized less than a false negative or a false positive.

Now, consider the algorithm segmentations shown in Figure 5. Each of the five algorithms makes a mistake on either the boundary between the first and second segment of the reference segmention, or within the second segment. How should these various segmentations be penalized? In the analysis below, we assume an application for which it is important not to introduce spurious boundaries. These comparisons will most likely vary depending on the goals of the target application.

Algorithm A-4 is arguably the worst of the examples, since it has a false positive and a false negative simultaneously. Algorithms A-0 and A-2 follow – they contain a

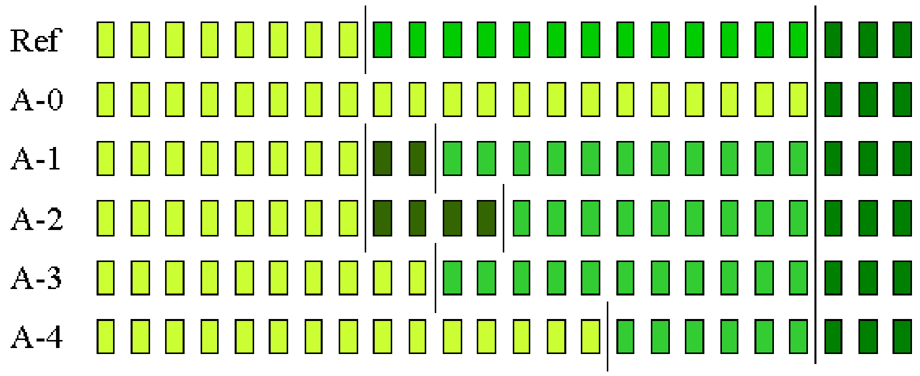

Figure 5
A reference segmentation and five different hypothesized segmentations with different properties.

pure false negative and false positive, respectively. Comparing algorithms A-1 and A-3, algorithm A-3 is arguably better, because it recognizes that there is only one boundary present rather than two. Algorithm A-1 does not recognize this, and inserts an extra segment. Even though algorithm A-1 actually places a correct boundary, it also places an erroneous boundary which, although close to the actual one, is still a false positive – in fact, a pure false positive. For this reason, algorithm A-3 can be considered better than algorithm A-1.

Now, consider how P_k treats the five types of mistakes above. Again, assume the first and second segments in the reference segmentation are average size segments. Algorithm A-4 is penalized the most, as it should be. The penalty is as much as $2k$ if the false positive falls in the middle of segment C, and is $> k$ as long as the false positive is a distance $> \frac{k}{2}$ from the actual boundary between the first and second reference segments. The penalty is large because the metric catches both the false negative and the false positive errors. Segmentations A-0 and A-2 are treated as discussed earlier in conjunction with Problem 1 – segmentation A-0 has a false negative, and thus has a penalty of k, and segmentation A-2 has a false positive, and thus incurs a penalty of $\leq k$. Finally, consider segmentations A-1 and A-3, and suppose that both contain an incorrect boundary some small distance e from the actual one. Then the penalty for algorithm A-1 is e, while the penalty for algorithm A-3 is $2e$. This should not be the case; algorithm A-1 should be penalized more than algorithm A-3, since a near-miss error is better than a pure false positive, even if it is close to the boundary.

Problem 5 - What do the numbers mean?

P_k is non-intuitive because it measures the probability that two sentences k units apart are incorrectly labelled as being in different segments, rather than directly reflecting the competence of the algorithm. Although perfect algorithms score 0, and various degenerate ones score 0.5, numerical interpretation and comparison are difficult because it is not clear how the scores are scaled.

3. A Solution

It turns out that a simple change to the error metric algorithm remedies most of the problems described above, while still retaining the desirable characteristic of penalizing near-misses less than pure false positives and pure false negatives. The fix, which we call WindowDiff, works as follows: for each position of the probe, simply compare how many reference segmentation boundaries fall in this interval (r_i) versus how many boundaries are assigned by the algorithm (a_i). The algorithm is penalized if $r_i \neq a_i$ (which is computed as $|r_i - a_i| > 0$).

More formally,

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

where $b(i,j)$ represents the number of boundaries between positions i and j in the text and N represents the number of sentences in the text.

This approach clearly eliminates the asymmetry between the false positive and false negative penalties seen in the P_k metric. It also catches false positives and false negatives within segments of length less than k.

To understand the behavior with respect to the other problems, consider again the examples in Figure 5. This metric penalizes algorithm A-4 (which contains both a false positive and a false negative) the most, assigning it a penalty of about $2k$. Algorithms A-0, A-1 and A-2 are assigned the same penalty (about k), and algorithm A-3 receives the smallest penalty ($2e$, where e is the offset from the actual boundary, presumed to be much smaller than k). Thus, although it makes the mistake of penalizing algorithm A-1 as much as algorithms A-0 and A-2, it correctly recognizes that the error made by algorithm A-3 is a near-miss, and assigns it a penalty less than algorithm A-1 or any of the others. We argue that this kind of error is less detrimental than the errors made by P_k. This metric successfully distinguishes the near-miss error as a separate kind of error, and penalizes it a different amount, something that P_k is unable to do.

We explored a weighted version of this metric, in which the penalty is weighted by the difference $|r_i - a_i|$. However, the results of the simulations were nearly identical with those of the non-weighted version of WindowDiff, so we do not consider the weighted version further.

4. Validation via Simulations

This section describes a set of simulations that verify the theoretical analysis of the P_k metric presented above, and also reports the results of simulating two alternatives, including the proposed solution just described.

For the simulation runs described below, three metrics were implemented:

- The P_k metric
- The P_k metric modified to double the false positive penalty (henceforth P'_k), and
- Our proposed alternative which counts the number of segment boundaries between the two ends of the probe, and assigns a penalty if this number is different for the experimental vs. the reference segmentation (henceforth WindowDiff, or WD).

In these studies, a single trial consists of generating a reference segmentation of 1000 segments with some distribution, generating different experimental segmentations of a specific type 100 times, computing the metric based on the comparison of the reference and the experimental segmentation, and averaging the 100 results. For example, we might generate a reference segmentation R, then generate 100 experimental segmentations that have false negatives with probability 0.5, and then compute the average of their P_k penalties. We carried out 10 such trials for each experiment, and averaged the average penalties over these trials.

4.1 Variation in the Segment Sizes

The first set of tests was designed to test the metric's performance on texts with different segment size distributions (Problem 3). We generated four sets of reference segmentations with segment size uniformly distributed between two numbers. Note that the units of segmentation are deliberately left unspecified. So a segment of size 25 can refer to 25 words, clauses, or sentences – whichever is applicable to the task under consideration. Also note that the same tests were run using larger segment sizes than those reported here, with the results remaining nearly identical.

For these tests, the mean segment size was held constant at 25 for each set of reference segments, in order to produce distributions of segment size with the same means but different variances. The four ranges of segment sizes were (20, 30), (15, 35), (10, 40), and (5, 45). The results of these tests are shown in Table 1. The tests used the following types of experimental segmentations:

- FN: segmentation with false negative probability 0.5 at each boundary

Table 1
Average error score for P_k, P'_k, and WD over 10 trials of 100 measurements each, shown by segment size distribution range. (a) False negatives were placed with probability 0.5 at each boundary, (b) false positives were placed with probability 0.5, uniformly distributed within each segment, and (c) both false negatives and false positives were placed with probability 0.5.

(a)	False Negatives, $p = 0.5$			
	(20, 30)	(15, 35)	(10, 40)	(5, 45)
P_k	0.245	0.245	0.240	0.223
P'_k	0.245	0.245	0.240	0.223
WD	0.245	0.245	0.242	0.237

(b)	False Positives, $p = 0.5$			
	(20, 30)	(15, 35)	(10, 40)	(5, 45)
P_k	0.128	0.122	0.112	0.107
P'_k	0.256	0.245	0.225	0.213
WD	0.240	0.241	0.238	0.236

(c)	False Positives and False Negatives, $p = 0.5$			
	(20, 30)	(15, 35)	(10, 40)	(5, 45)
P_k	0.317	0.309	0.290	0.268
P'_k	0.446	0.432	0.403	0.375
WD	0.376	0.370	0.357	0.343

- FP: segmentation with false positive probability 0.5 in each segment, with the probability uniformly distributed within each segment
- FNP: segmentation with false positive probability 0.5 (uniformly distributed), and false negative probability of 0.5.

The results indicate that variation in segment size does make a difference, but not a very big one. (As will be seen below, the differences are similar when we use a smaller probability of false negative/positive occurrence.) The P_k value for the (20, 30) range with FN segmentation is on average 0.245, and decreases to 0.223 for the (5, 45) range. Similarly, the FP segmentation decreases from 0.128 for the (20, 30) range to 0.107 for the (5, 45) range, and the FNP segmentation decreases from 0.317 for the (20, 30) range to 0.268 for the (5, 45) range. Thus, variation in segment size has an effect on P_k, as predicted.

Note that for false negatives, the P_k value for the (20, 30) range is not much different than for the (15, 35) range. This is expected since there are no segments of size less than k (12.5) in these conditions. For the (10, 40) range the P_k value is slightly smaller, and for the (5, 45) range it is smaller still. These results are to be expected, since more segments in these ranges will be of length less than k.

For the FP segmentations, on the other hand, the decrease in P_k is more pronounced, decreasing from 0.128 to 0.107 as the segment size range changes from (20, 30) to (5, 45). This is also consistent with our earlier analysis of the behavior of the metric on false positives as segment size decreases. Notice that the difference in P_k between (15, 35) and (10, 40) is slightly larger than the other two differences. This happens because for segment sizes $< k$, the false positive penalty disappears completely. The results for the FNP segmentation is consistent with what one would expect of a mix of the FN and FP segmentations.

Several other observations can be made from Table 1. We can begin to make some judgments as to how the metric performs on algorithms prone to different kinds of errors. First, P_k penalizes false negatives about twice as much as false positives, as predicted by our analysis. The experimental segmentations in Table 1a contain on average 500 false negatives, while the ones in Table 1b contain on average 500 false positives, but the penalty for the Table 1b segmentations is consistently about half that of Table 1a. Thus, algorithms prone to false positives are penalized less harshly than those prone to false negatives.

The table also shows the performance of the two other metrics. P'_k simply doubles false positive penalty, while WD counts and compares the number of boundaries between the two ends of the probe, as described earlier. Both P'_k and WD appear to solve the problem of under-penalizing false positives, but WD also has the added benefit of being more stable across variations in segment size distribution. Thus, WD essentially solves Problems 1, 2 and 3.

Table 1c shows that for the FNP segmentation (in which both false positives and false negatives occur) there is a disparity between the performances of P'_k and WD. It appears that P'_k is harsher in this situation. From the above discussion, we know that WD is more lenient in situations where there is a false negative and a false positive near each other (where "near" means within a distance of $\frac{k}{2}$) than P'_k is. However, P'_k is more lenient for pure false positives that occur close to boundaries. Thus, it is not immediately clear why P'_k is harsher in this situation, but a more detailed look provides the answer.

Let us begin the analysis by trying to explain why P_k scores for the FNP segmentation make sense. The FNP segmentation places both false negatives and false positives with probability 0.5. Since we are working with reference segmentations of 1000 segments, this means 500 missed boundaries and 500 incorrect boundaries. Since the probabilities are uniformly distributed across all segments and all boundaries, on average one would expect the following distribution of errors:

- 250 false positives with no false negative within k sentences of them (Type A),
- 250 false negatives with no false positive within k sentences of them (Type B), and
- 250 "joint" errors, where a false positive and a false negative occur within k sentences of each other (Type C).

A Type A error is a standard false positive, so the average penalty is $\frac{k}{2}$. A Type B error is a standard false negative, so the average penalty is k. It remains to figure out what the average penalty is for a Type C error. Modeling the behavior of the metric, a Type C error occurrence in which a false positive and a false negative are some distance $e < k$ from each other incurs a penalty of $2e$, where e is assigned for the false positive and another e is assigned for the false negative. This may range from 0 to $2k$, and since error distribution is uniform, the penalty is k on average – the same as for a regular false negative. To translate this into actual values, we assume the metric is linear with respect to the number of errors (a reasonable assumption, supported by our experiments). Thus, if P_k outputs a penalty of p for 500 false negatives, it would have a penalty of $\frac{p}{2}$ for 250 false negatives. Letting a be the penalty for 500 Type A errors, b the penalty for 500 Type B errors and c the penalty for 500 Type C errors, we get that the penalty for the FNP segmentation is $p = \frac{a}{2} + \frac{b}{2} + \frac{c}{2}$. Assuming the metric is linear, we know that $c = b = 2a$ (because P_k penalized false negatives twice as much as false positives on average). We can thus substitute either b or $2a$ for c. We choose to substitute $2a$, because P_k is strongly affected by segment size variation for Type A and Type C errors, but not for Type B errors. Thus, replacing c with $2a$ is more accurate. Performing the substitution, we have $p = 3 \cdot \frac{a}{2} + \frac{b}{2}$. We have a and b from the FP and FN data, respectively, so we can compute p. The results, arranged by segment size variation, are as follows:

	(20, 30)	(15, 35)	(10, 40)	(5, 45)
Estimate	0.315	0.306	0.288	0.272
Actual	0.317	0.309	0.290	0.268

As can easily be seen, the estimate produced using this method is very similar to the actual P_k value.

The same sort of analysis applies for P'_k and WD. In P'_k, Type A errors are penalized k on average, since false positive penalty is doubled. Type B errors have an average penalty of k, as for P_k. Type C errors have an average penalty of $3e$, where $2e$ is assigned for the false positive and e is assigned for the false negative. This means that the average penalty for a Type C error is $3 \cdot \frac{k}{2}$. Since we know that $c = 1.5a$ by the linear metric assumption, we have $p = \frac{a}{2} + \frac{b}{2} + 1.5 \cdot \frac{a}{2} = 5 \cdot \frac{a}{4} + \frac{b}{2}$ (the choice of $1.5a$ over $1.5b$ was made for the same reason as the choice of $2a$ over b in the calculations for P_k). The results, arranged by segment size variation, are as follows:

	(20, 30)	(15, 35)	(10, 40)	(5, 45)
Estimate	0.443	0.429	0.401	0.378
Actual	0.446	0.432	0.403	0.375

Finally, WD incurs an average penalty of k for both Type A and Type B errors. For Type C errors, the penalty is $2e$, so it is also k on average. Thus we get $p = \frac{a}{2} + \frac{b}{2} + \frac{a}{2} = a + \frac{b}{2}$. The results, arranged by segment size variation, are as follows:

	(20, 30)	(15, 35)	(10, 40)	(5, 45)
Estimate	0.363	0.364	0.359	0.355
Actual	0.376	0.370	0.357	0.343

Table 2
Average error score for P_k, P'_k, and WD over 10 trials of 100 measurements each, shown by segment size distribution range. (a) False negatives were placed with probability 0.05 at each boundary, (b) false positives were placed with probability 0.05, uniformly distributed within each segment, and (c) both false negatives and false positives were placed with probability 0.05. (d) False negatives were placed with probability 0.25 at each boundary, (e) false positives were placed with probability 0.25, uniformly distributed within each segment, and (f) both false negatives and false positives were placed with probability 0.25.

(a)	False Negatives, $p = 0.05$			
	(20, 30)	(15, 35)	(10, 40)	(5, 45)
P_k	0.025	0.025	0.024	0.022
P'_k	0.025	0.025	0.024	0.022
WD	0.025	0.025	0.024	0.024

(b)	False Positives, $p = 0.05$			
	(20, 30)	(15, 35)	(10, 40)	(5, 45)
P_k	0.013	0.012	0.011	0.011
P'_k	0.026	0.025	0.023	0.021
WD	0.024	0.024	0.024	0.024

(c)	False Positives and False Negatives, $p = 0.05$			
	(20, 30)	(15, 35)	(10, 40)	(5, 45)
P_k	0.037	0.036	0.035	0.032
P'_k	0.050	0.048	0.046	0.042
WD	0.048	0.048	0.048	0.047

(d)	False Negatives, $p = 0.25$			
	(20, 30)	(15, 35)	(10, 40)	(5, 45)
P_k	0.122	0.122	0.121	0.110
P'_k	0.122	0.122	0.121	0.110
WD	0.122	0.122	0.122	0.121

(e)	False Positives, $p = 0.25$			
	(20, 30)	(15, 35)	(10, 40)	(5, 45)
P_k	0.064	0.061	0.056	0.053
P'_k	0.129	0.123	0.112	0.106
WD	0.121	0.121	0.121	0.120

(f)	False Positives and False Negatives, $p = 0.25$			
	(20, 30)	(15, 35)	(10, 40)	(5, 45)
P_k	0.172	0.168	0.161	0.147
P'_k	0.236	0.229	0.217	0.200
WD	0.215	0.213	0.211	0.205

These estimates don't correspond to the actual results quite as closely as the estimates for P_k and P'_k did, but they are still very close. One of the reasons that these estimates are a little less accurate is that for WD, Type C errors are more affected by variation in segment size than either Type A or Type B errors. This is clear from the greater decrease in the actual data than in the estimate.

Table 2 shows data similar to that of Table 1, but using two different probability values for error occurrence: 0.05 and 0.25. These results have the same tendencies as those shown above for $p = 0.5$.

4.2 Variation in the Error Distributions

The second set of tests was designed to assess the performance of the metrics on algorithms prone to different kinds of errors. This would determine whether the metrics are consistent in their applications of penalty, or whether they favor certain kinds of errors over others. For these trials, we generated the reference segmentation using a uniform distribution of segment sizes in the (15, 35) range. We picked this range because it has reasonably high segment size variation, but segment size does not dip below k. For the reasons described above, this means the results will not be skewed by the sensitivity of P_k and P'_k to segment size variations.

The tests analyzed below were performed using the high error occurrence probabilities of 0.5, but similar results were obtained using probabilities of 0.25 and 0.05 as well. The following error distributions were used:[1]

- FN: False negatives, probability $p = 0.5$
- FP1: False positives uniformly distributed in each segment, probability $p = 0.5$
- FP2: False positives normally distributed around each boundary with standard deviation equal to $\frac{1}{4}$ the segment size, probability $p = 0.5$
- FP3: False positives uniformly distributed throughout the document, occurring at each point with probability $p = \frac{numberofsegments}{length \cdot 2}$. This corresponds to a 0.5 probability value for each individual segment.
- FNP1: FN and FP1 combined
- FNP2: FN and FP2 combined
- FNP3: FN and FP3 combined

The results are shown in Table 3. P_k penalizes FP2 less than FP1 and FP3, and FNP2 less than FNP1 and FNP3. This result is as expected. FP2 and FNP2 have false positives normally distributed around each boundary, which means that more of the false positives are close to the boundaries, and thus are penalized less. If we made the standard deviation smaller, we would expect this difference to be even more apparent.

P'_k penalized FP2 and FNP2 the least in their respective categories, and FP1 and FNP1 the most, with FP3 and FNP3 falling in between. These results are as expected, for the same reasons as for P_k. The difference in the penalty for FP1 and FP3 (and FNP1 vs. FNP3) – both for P_k and P'_k, but especially apparent for P'_k – is interesting. In FP/FNP1, false positive probability is uniformly distributed throughout each segment, whereas in FP/FNP3, false positive probability is uniformly distributed throughout the entire

1 Normal distributions were calculated using the gaussrand() function from (Box and Muller, 1958), found online at http://www.eskimo.com/∼scs/C-faq/q13.20.html.

Table 3
Average error score for P_k, P'_k, and WD over 10 trials of 100 measurements each over the segment distribution range (15, 35) and with error probabilities of 0.5. The average penalties computed by the three metrics are shown for seven different error distributions.

	FN	FP1	FP2	FP3	FNP1	FNP2	FNP3
P_k	.245	.122	.091	.112	.308	.267	.304
P'_k	.245	.244	.182	.224	.431	.354	.416
WD	.245	.240	.236	.211	.370	.341	.363

document. Thus, the FP/FNP3 segmentations are more likely to have boundaries that are very close to each other, since they are not segment-dependent, while FP/FNP1 are limited to at most one false positive per segment. This results in P'_k assigning smaller penalties for FP/FNP3, since groups of false positives close together (to be more exact, within k sentences of each other) would be under-penalized. This difference is also present in the P_k results, but is about half for obvious reasons.

WD penalized FP1 the most and FP3 the least among the FP segmentations. Among the FNP segmentations, FNP1 was penalized the most, and FNP2 the least. To see why, we examine the results for the FP segmentations. WD penalizes pure false positives the same amount regardless of how close they are to a boundary; the only way false positives are underpenalized is if they occur in bunches. As mentioned earlier, this is most likely to happen in FP3. It is least likely to happen in FP1, since in FP1 there is a maximum of one false positive per segment, and this false positive is not necessarily close to a boundary. In FP2, false positives are also limited to one per segment, but they are also more likely to be close to boundaries. This increases the likelihood of 2 false positives being within k sentences of each other, and thus makes WD give a slightly lower score to the FP2 segmentation than to the FP1 segmentation.

Now let us move on to the FNP segmentations. FNP3 is penalized less than FNP1 for the same reason as described above, and FNP2 is penalized even less than FNP3. The closer a Type C error is to the boundary, the lower the penalty. FNP2 has more errors distributed near the boundaries than the others, thus the FNP2 segmentation is penalized less than either FNP1 or FNP3.

The same tests were run for different error occurrence probabilities ($p = 0.05$ and $p = 0.25$), achieving similar results to those for $p = 0.5$ just described. There is a slight difference for the case of $p = 0.05$ because the error probability is too small for some of the trends to manifest themselves. In particular, the differences in the way WD treats the different segmentations disappear when the error probability is this small.

4.3 Variation in the Error Types

We also performed a small set of tests to verify the theoretical finding that P_k and P'_k overpenalize near-miss errors as compared to pure false positives, and WD does the opposite – it overpenalizes the pure false positives. Space limitations prevent detailed reporting of these results, but the simulations did indeed verify these expectations.

5. Conclusions

We have found that P_k error metric for text segmentation algorithms is affected by the variation of segment size distribution, becoming slightly more lenient as the variance increases. It penalizes false positives significantly less than false negatives, particularly if the false positives are uniformly distributed throughout the document. It penalizes near-miss errors more than pure false positives of equal magnitude. Finally, it fails to take into account situations in which multiple boundaries occur between the two sides of the probe, and often misses or under-penalizes mistakes in small segments.

We proposed two modifications to tackle these problems. The first, which we called P_k', simply doubles the false positive penalty. This solves the problem of overpenalizing false negatives, but is not effective at dealing with the other problems. The second, which we call WindowDiff (WD), counts the number of boundaries between the two ends of a fixed length probe, and compares this number to the number of boundaries found in the same window of text for the reference segmentation. This modification addresses all of the problems listed above. WD is only slightly affected by variation of segment size distribution, gives equal weight to the false positive penalty with the false negative penalty, is able to catch mistakes in small segments just as well as mistakes in large segments, and penalizes near-miss errors less than pure false positives of equal magnitude. However, it has some problems of its own. WD penalizes all pure false positives the same amount regardless of how close they are to an actual boundary. It is not clear whether this is a good thing or not, but it seems to be preferable to over-penalizing near-misses.

The discussion above addresses Problems 1 through 4 but does not address Problem 5: how does one interpret the values produced by the metric? From the tests we have run, it appears that the WD metric grows in a roughly linear fashion with the difference between the reference and the experimental segmentations. In addition, we feel that WD is a more meaningful metric than P_k. Comparing two stretches of text to see how many discrepancies occur between the reference and the algorithm's result seems more intuitive than determining how often two text units are incorrectly labelled as being in different segments.

Acknowledgments

This work was completed while the second author was a visiting professor at Harvard University. Both authors thank Barbara Grosz and Stuart Shieber, without whom this work would not have happened, and Freddy Choi for some helpful explanations. They would also like to thank the anonymous reviewers for their valuable comments.

Partial support for the research reported in this paper was provided by National Science Foundation grants IRI-9618848 and CDA-94-01024.

References

Allan, James, Jaime Carbonell, George Doddington, Jonathan Yamron, and Yiming. Yang. 1998. Topic detection and tracking pilot study: Final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, Lansdowne, VA, February.

Baeza-Yates, Ricardo and Berthier Ribeiro-Neto. 1999. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Company.

Barzilay, Regina and Michael Elhadad. 1997. Using lexical chains for text summarization. In *Proceedings of the ACL Intelligent Scalable Text Summarization Workshop (ISTS'97)*.

Beeferman, Douglas, Adam Berger, and John Lafferty. 1997. Text segmentation using exponential models. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 35–46.

Beeferman, Douglas, Adam Berger, and John Lafferty. 1999. Statistical models of text segmentation. *Machine Learning*, 34(1-3), February.

Boguraev, Branimir and Mary Neff. 2000. Discourse segmentation in aid of document summarization. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, Maui, HI, January.

Box, G.E.P. and M. E. Muller. 1958. A note in the generation of random normal deviates. *Annals. Math. Stat.*, 29:610–611.

Callan, James P. 1994. Passage-level evidence in document retrieval. In *Proceedings of the 17th Annual International ACM/SIGIR Conference*, pages 302–310, Dublin, Ireland.

Choi, Freddy. 2000. Advances in domain independent linear text segmentation. In *Proceedings of the NAACL'00*, May.

Dharanipragada, S., M. Franz, Jeffrey S. McCarley, S. Roukos, and Todd Ward. 1999. Story segmentation and topic detection in the broadcast news domain. In *The Proceedings of the DARPA Broadcast News Workshop*, Feb 28 - March 3.

Eichmann, David, Miguel Ruiz, Padmini Srinivasan, Nick Street, Chris Culy, and Filippo Menczer. 1999. A cluster-based approach to tracking, detection and segmentation of broadcast news. In *The Proceedings of the DARPA Broadcast News Workshop*, Feb 28 - March 3.

Hasnah, Ahmad. 1996. *Full Text Processing and Retrieval: Weight Ranking, Text Structuring, and Passage Retrieval for Arabic Documents*. Ph.D. thesis, Illinois Institute of Technology.

Hauptmann, Alexander G. and Michael J. Whitbrock. 1998. Story segmentation and detection of commercials in broacast news video. In *Proceedings of the ADL-98 Advances in Digital Libraries*, Santa Barbara, CA, April.

Hearst, Marti A. 1993. TextTiling: A quantitative approach to discourse segmentation. Technical Report Sequoia 93/24, Computer Science Division, University of California, Berkeley.

Hearst, Marti A. 1994. Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd Meeting of the Association for Computational Linguistics*, pages 9–16, Las Cruces, NM, June.

Hearst, Marti A. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64, March.

Hearst, Marti A. and Christian Plaunt. 1993. Subtopic structuring for full-length document access. In *Proceedings of the 16th Annual International ACM/SIGIR Conference*, pages 59–68, Pittsburgh, PA.

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

Hirschberg, Julia and Christine H. Nakatani. 1996. A prosodic analysis of discourse segments in direction-giving monologues. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 286–293, Santa Cruz, CA.

Kan, Min-Yen, Judith L. Klavans, and Kathleen R. Mckeown. 1998. Linear segmentation and segment significance. In *Sixth Workshop on Very Large Corpora*. Association for Computational Linguistics.

Karlgren, Jussi. 1996. Stylistic variation in an information retrieval experiment. In *Proceedings of the NeMLaP-2 Conference*, Ankara, Turkey, September.

Kaszkiel, Marcin and Justin Zobel. 1997. Passage retrieval revisited. In *Proceedings of the Twentieth International Conference on Research and Development in Information Access (ACM SIGIR)*, pages 178–185.

Litman, Diane J. and Rebecca J. Passonneau. 1995. Combining multiple knowledge sources for discourse segmentation. In *Proceedings of the 33rd Meeting of the Association for Computational Linguistics*, pages 108–115, June.

Mani, Inderjeet, David House, Mark Maybury, and Morgan Green. 1997. Towards content-based browsing of broadcast news video. In Mark Maybury, editor, *Intelligent Multimedia Information Retrieval*. AAAI/MIT Press.

Manning, Christopher D. 1998. Rethinking text segmentation models: An information extraction case study. Technical Report SULTRY-98-07-01, University of Syndey.

Marcu, Daniel. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press.

Merlino, Andy, Daryl Morey, and Mark Maybury. 1997. Broadcast news navigation using story segmentation. In *Proceedings of ACM Multimedia*, pages 157–164, November.

Mittal, Vibhu, Mark Kantrowitz, Jade Goldstein, and Jaime Carbonell. 1999. Selecting text spans for document summaries: Heuristics and metrics. In *Proceedings of 16th Annual Conference on Artificial Intelligence (AAAI 99)*, Orlando, FL.

Moffat, Alistair, Ron Sacks-Davis, Ross Wilkinson, and Justin Zobel. 1994. Retrieval of partial documents. In Donna Harman, editor, *Proceedings of the Second Text Retrieval Conference TREC-2*. National Institute of Standards and Technology Special Publication 500-215, pages 181–190.

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

Nomoto, Tadashi and Yoshihiko Nitta. 1994. A grammatico-statistical approach to discourse partitioning. In *Proceedings of the Fifteenth International Conference on Computational Linguistics (COLING)*, pages 1145–1150, Kyoto, Japan, August.

Passonneau, Rebecca J. and Diane J. Litman. 1993. Intention-based segmentation: Human reliability and correlation with linguistic cues. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 148–155.

Ponte, Jay and Bruce Croft. 1997. Text segmentation by topic. In *Proceedings of the First European Conference on Research and Advanced Technology for Digitial Libraries*.

Reynar, Jeffrey C. 1994. An automatic method of finding topic boundaries. In *Proceedings of the Student Session of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 331–333, Las Cruces, NM.

Richmond, Korin, Andrew Smith, and Einat Amitay. 1997. Detecting subject boundaries within text: A language independent statistical approach. In *Proceedings of Second Conference on Empirical Methods in Natural Language Processing*, pages 47–54. ACL.

Salton, Gerard, James Allan, and Chris Buckley. 1993. Approaches to passage retrieval in full text information systems. In *Proceedings of the 16th Annual International ACM/SIGIR Conference*, pages 49–58, Pittsburgh, PA.

Salton, Gerard, James Allan, Chris Buckley, and A. Singhal. 1994. Automatic analysis, theme generation, and summarization of machine-readable texts. *Science*, 264(5164):1421–1426, Jun.

van Mulbregt, P., L. Gillick I. Carp, S. Lowe, and J. Yamron. 1999. Segmentation of automatically transcribed broadcast news text. In *The Proceedings of the DARPA Broadcast News Workshop*, Feb 28 - March 3.

Yaari, Yaakov. 1997. Segmentation of expository text by hierarchical agglomerative clustering. In *Recent Advances in NLP (RANLP'97)*, Bulgaria.