Segmentation Similarity and Agreement

Chris Fournier
University of Ottawa
Ottawa, ON, Canada
cfour037@eecs.uottawa.ca

Diana Inkpen
University of Ottawa
Ottawa, ON, Canada
diana@eecs.uottawa.ca

Abstract

We propose a new segmentation evaluation metric, called *segmentation similarity* (S), that quantifies the similarity between two segmentations as the proportion of boundaries that are not transformed when comparing them using edit distance, essentially using edit distance as a penalty function and scaling penalties by segmentation size. We propose several adapted inter-annotator agreement coefficients which use S that are suitable for segmentation. We show that S is configurable enough to suit a wide variety of segmentation evaluations, and is an improvement upon the state of the art. We also propose using inter-annotator agreement coefficients to evaluate automatic segmenters in terms of human performance.

1 Introduction

Segmentation is the task of splitting up an item, such as a document, into a sequence of segments by placing boundaries within. The purpose of segmenting can vary greatly, but one common objective is to denote shifts in the topic of a text, where multiple boundary types can also be present (*e.g.*, major versus minor topic shifts). Human-competitive automatic segmentation methods can help a wide range of computational linguistic tasks which depend upon the identification of segment boundaries in text.

To evaluate automatic segmentation methods, a method of comparing an automatic segmenter's performance against the segmentations produced by human judges (coders) is required. Current methods of performing this comparison designate only one coder's segmentation as a reference to compare against. A single "true" reference segmentation from a coder should not be trusted, given that inter-annotator agreement is often reported to be rather poor (Hearst, 1997, p. 54). Additionally, to ensure that an automatic segmenter does not over-fit to the preference and bias of one particular coder, an automatic segmenter should be compared directly against multiple coders.

The state of the art segmentation evaluation metrics (P_k and WindowDiff) slide a window across a designated reference and hypothesis segmentation, and count the number of windows where the number of boundaries differ. Window-based methods suffer from a variety of problems, including: i) unequal penalization of error types; ii) an arbitrarily defined window size parameter (whose choice greatly affects outcomes); iii) lack of clear intuition; iv) inapplicability to multiply-coded corpora; and v) reliance upon a "true" reference segmentation.

In this paper, we propose a new method of comparing two segmentations, called *segmentation similarity* [1] (S), that: i) equally penalizes all error types (unless explicitly configured otherwise); ii) appropriately responds to scenarios tested; iii) defines no arbitrary parameters; iv) is intuitive; and v) is adapted for use in a variety of popular inter-annotator agreement coefficients to handle multiply-coded corpora; and vi) does not rely upon a "true" reference segmentation (it is symmetric). Capitalizing on the adapted inter-annotator agreement coefficients, the relative difficulty that human segmenters have with various segmentation tasks can now be quantified. We also propose that these coefficients can be used to evaluate and compare automatic segmentation methods in terms of human agreement.

This paper is organized as follows. In Section 2, we review segmentation evaluation and inter-annotator agreement. In Section 3, we present S and

[1] A software implementation of segmentation similarity (S) is available at http://nlp.chrisfournier.ca/

inter-annotator agreement coefficient adaptations. In Section 4, we evaluate S and WindowDiff in various scenarios and simulations, and upon a multiply-coded corpus.

2 Related Work

2.1 Segmentation Evaluation

Precision, recall, and their mean (F_β-measure) have been previously applied to segmentation evaluation. Precision is the proportion of boundaries chosen that agree with a reference segmentation, and recall is the proportion of boundaries chosen that agree with a reference segmentation out of all boundaries in the reference and hypothesis (Pevzner and Hearst, 2002, p. 3). For segmentation, these metrics are unsuitable because they penalize near-misses of boundaries as full-misses, causing them to drastically overestimate the error. Near-misses are prevalent in segmentation and can account for a large proportion of the errors produced by a coder, and as inter-annotator agreement often shows, they do not reflect coder error, but the difficulty of the task.

P_k (Beeferman and Berger, 1999, pp. 198–200)[2] is a window-based metric which attempts to solve the harsh near-miss penalization of precision, recall, and F_β-measure. In P_k, a window of size k, where k is defined as half of the mean reference segment size, is slid across the text to compute penalties. A penalty of 1 is assigned for each window whose boundaries are detected to be in different segments of the reference and hypothesis segmentations, and this count is normalized by the number of windows.

Pevzner and Hearst (2002, pp. 5–10) highlighted a number of issues with P_k, specifically that: i) False negatives (FNs) are penalized more than false positives (FPs); ii) It does not penalize FPs that fall within k units of a reference boundary; iii) Its sensitivity to variations in segment size can cause it to linearly decrease the penalty for FPs if the size of any segments fall below k; and iv) Near-miss errors are too harshly penalized.

To attempt to mitigate the shortcomings of P_k, Pevzner and Hearst (2002, p. 10) proposed a modified metric which changed how penalties were counted, named *WindowDiff* (WD). A window of size k is still slid across the text, but now penalties are attributed to windows where the number of boundaries in each segmentation differs (see Equation 1, where $\mathrm{b}(R_{ij})$ and $\mathrm{b}(H_{ij})$ represents the number of boundaries within the segments in a window of size k from position i to j, and N the number of sentences plus one), with the same normalization.

$$\mathrm{WD}(R,H) = \frac{1}{N-k} \sum_{i=1,j=i+k}^{N-k} (|\mathrm{b}(R_{ij}) - \mathrm{b}(H_{ij})| > 0) \quad (1)$$

WindowDiff is able to reduce, but not eliminate, sensitivity to segment size, gives more equal weights to both FPs and FNs (FNs are, in effect, penalized less[3]), and is able to catch mistakes in both small and large segments. It is not without issues though; Lamprier et al. (2007) demonstrated that WindowDiff penalizes errors less at the beginning and end of a segmentation (this is corrected by padding the segmentation at each end by size k). Additionally, variations in the window size k lead to difficulties in interpreting and comparing WindowDiff's values, and the intuition of the method remains vague.

Franz et al. (2007) proposed measuring performance in terms of the number of words that are FNs and FPs, normalized by the number of word positions present (see Equation 2).

$$R_{FN} = \frac{1}{N}\sum_w FN(w), \quad R_{FP} = \frac{1}{N}\sum_w FP(w) \quad (2)$$

R_{FN} and R_{FP} have the advantage that they take into account the severity of an error in terms of segment size, allowing them to reflect the effects of erroneously missing, or added, words in a segment better than window based metrics. Unfortunately, R_{FN} and R_{FP} suffer from the same flaw as precision, recall, and F_β-measure in that they do not account for near misses.

2.2 Inter-Annotator Agreement

The need to ascertain the agreement and reliability between coders for segmentation was recognized

[2] P_k is a modification of P_μ (Beeferman et al., 1997, p. 43). Other modifications such as TDT C_{seg} (Doddington, 1998, pp. 5–6) have been proposed, but P_k has seen greater usage.

[3] Georgescul et al. (2006, p. 48) note that both FPs and FNs are weighted by $1/N-k$, and although there are "equiprobable possibilities to have a [FP] in an interval of k units", "the total number of equiprobable possibilities to have a [FN] in an interval of k units is smaller than $(N-k)$", making the interpretation of a full miss as a FN less probable than as a FP.

by Passonneau and Litman (1993), who adapted the percentage agreement metric by Gale et al. (1992, p. 254) for usage in segmentation. This percentage agreement metric (Passonneau and Litman, 1993, p. 150) is the ratio of the total observed agreement of a coder with the majority opinion for each boundary over the total possible agreements. This measure failed to take into account chance agreement, or to less harshly penalize near-misses.

Hearst (1997) collected segmentations from 7 coders while developing the automatic segmenter TextTiling, and reported mean κ (Siegel and Castellan, 1988) values for coders and automatic segmenters (Hearst, 1997, p. 56). Pairwise mean κ scores were calculated by comparing a coder's segmentation against a reference segmentation formulated by the majority opinion strategy used in Passonneau and Litman (1993, p. 150) (Hearst, 1997, pp. 53–54). Although mean κ scores attempt to take into account chance agreement, near misses are still unaccounted for, and use of Siegel and Castellan's (1988) κ has declined in favour of other coefficients (Artstein and Poesio, 2008, pp. 555–556).

Artstein and Poesio (2008) briefly touch upon recommendations for coefficients for segmentation evaluation, and though they do not propose a measure, they do conjecture that a modification of a weighted form of α (Krippendorff, 1980; Krippendorff, 2004) using unification and WindowDiff may suffice (Artstein and Poesio, 2008, pp. 580–582).

3 Segmentation Similarity

For discussing segmentation, a segment's size (or mass) is measured in units, the error is quantified in potential boundaries (PBs), and we have adopted a modified form of the notation used by Artstein and Poesio (2008), where the set of:

- *Items* is $\{i|i \in I\}$ with cardinality **i**;
- *Categories* is $\{k|k \in K\}$ with cardinality **k**;
- *Coders* is $\{c|c \in C\}$ with cardinality **c**;
- *Segmentations* of an item i by a coder c is $\{s|s \in S\}$, where when s_{ic} is specified with only one subscript, it denotes s_c, for all relevant items (i); and
- *Types* of segmentation boundaries is $\{t|t \in T\}$ with cardinality **t**.

3.1 Sources of Dissimilarity

Linear segmentation has three main types of errors:

1. s_1 contains a boundary that is off by n PBs in s_2;
2. s_1 contains a boundary that s_2 does not; or
3. s_2 contains a boundary that s_1 does not.

These types of errors can be seen in Figure 1, and are conceptualized as a pairwise *transposition* of a boundary for error 1, and the insertion or deletion (depending upon your perspective) of a boundary for errors 2 and 3. Since we do not designate either segmentation as a reference or hypothesis, we refer to insertions and deletions both as *substitutions*.

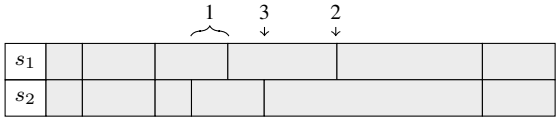


Figure 1: Types of segmentations errors

It is important to not penalize near misses as full misses in many segmentation tasks because coders often agree upon the existence of a boundary, but disagree upon its exact location. In the previous scenario, assigning a full miss would mean that even a boundary loosely agreed-upon, as in Figure 1, error 1, would be regarded as completely disagreed-upon.

3.2 Edit Distance

In S, concepts from Damereau-Levenshtein edit distance (Damereau, 1964; Levenshtein, 1966) are applied to model segmentation edit distance as two operations: substitutions and transpositions.[4] These two operations represent full misses and near misses, respectively. Using these two operations, a new globally-optimal minimum edit distance is applied to a pair of sequences of sets of boundaries to model the sources of dissimilarity identified earlier.[5]

Near misses that are remedied by transposition are penalized as b PBs of error (where b is the number of boundaries transposed), as opposed to the $2b$ PBs of errors by which they would be penalized if they were considered to be two separate substitution operations. Transpositions can also be considered over $n > 2$ PBs (n-wise transpositions). This is useful if, for a specific task, near misses of up to n PBs are not to be penalized as full misses (default $n = 2$).

The error represented by the two operations can also be scaled (*i.e.*, weighted) from 1 PB each to a

[4] Beeferman et al. (1997, p. 42) briefly mention using an edit distance without transpositions, but discard it in favour of P_μ.

[5] For multiple boundaries, an *add/del* operation is added, and transpositions are considered only within boundary types.

fraction. The distance over which an n-wise transposition occurred can also be used in conjunction with the scalar operation weighting so that a transposition is weighted using the function in Equation 3.

$$\text{te}(n, b) = b - ({}^{1}/_{b})^{n-2} \quad \text{where } n \geq 2 \text{ and } b > 0 \qquad (3)$$

This transposition error function was chosen so that, in an n-wise transposition where $n = 2$ PBs and the number of boundaries transposed $b = 2$, the penalty would be 1 PB, and the maximum penalty as $\lim_{n\to\infty} \text{te}(n)$ would be b PBs, or in this case 2 PBs (demonstrated later in Figure 5b).

3.3 Method

In S, we conceptualize the entire segmentation, and individual segments, as having mass (*i.e.*, unit magnitude/length), and quantify similarity between two segmentations as the proportion of boundaries that are not transformed when comparing segmentations using edit distance, essentially using edit distance as a penalty function and scaling penalties by segmentation size. S is a symmetric function that quantifies the similarity between two segmentations as a percentage, and applies to any granularity or segmentation unit (*e.g.*, paragraphs, sentences, clauses, etc.).

Consider a somewhat contrived example containing–for simplicity and brevity–only one boundary type ($\mathbf{t} = 1$). First, a segmentation must be converted into a sequence of segment mass values (see Figure 2).

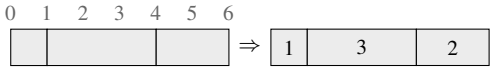


Figure 2: Annotation of segmentation mass

Then, a pair of segmentations are converted into parallel sequences of boundary sets, where each set contains the types of boundaries present at that potential boundary location (if there is no boundary present, then the set is empty), as in Figure 3.

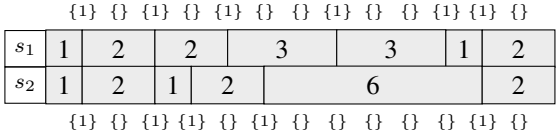


Figure 3: Segmentations annotated with mass and their corresponding boundary set sequences

The edit distance is calculated by first identifying all potential substitution operations that could occur (in this case 5). A search for all potential n-wise transpositions that can be made over n adjacent sets between the sequences is then performed, searching from the beginning of the sequence to the end, keeping only those transpositions which do not overlap and which result in transposing the most boundaries between the sequences (to minimize the edit distance). In this case, we have only one non-overlapping 2-wise transposition. We then subtract the number of boundaries involved in transpositions between the sequences (2 boundaries) from the number of substitutions, giving us an edit distance of 4 PBs: 1 transposition PB and 3 substitution PBs.

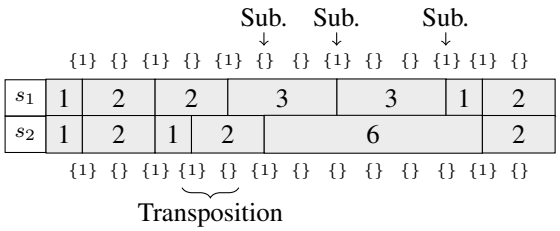


Figure 4: Edit operations performed on boundary sets

Edit distance, and especially the number of operations of each type performed, is useful in identifying the number of full and near misses that have occurred–which indicates whether one's choice of transposition window size n is either too generous or too harsh. Edit distance as a penalty does not incorporate information on the severity of an error with respect to the size of a segment, and is not an easily comparable value without some form of normalization. To account for these issues, we define S so that boundary edit distance is used to subtract penalties for each edit operation that occurs, from the number of potential boundaries in a segmentation, normalizing this value by the total number of potential boundaries in a segmentation.

$$\text{S}(s_{i1}, s_{i2}) = \frac{\mathbf{t} \cdot \text{mass}(i) - \mathbf{t} - \text{d}(s_{i1}, s_{i2}, T)}{\mathbf{t} \cdot \text{mass}(i) - \mathbf{t}} \qquad (4)$$

S, as shown in Equation 4, scales the mass of the item by the cardinality of the set of boundary types ($\mathbf{t}$) because the edit distance function $\text{d}(s_{i1}, s_{i1}, T)$ will return a value of $[0, \mathbf{t} \cdot \text{mass}(i)]$ PBs, where $\mathbf{t} \in \mathbb{Z}^{+}$–while subtracting the edit distance and $\mathbf{t}$.[6]

[6]The number of potential boundaries in a segmentation s_i

The numerator is normalized by the total number of potential boundaries per boundary type. This results in a function with a range of $[0, 1]$. It returns 0 when one segmentation contains no boundaries, and the other contains the maximum number of possible boundaries. It returns 1 when both segmentations are identical.

Using the default configuration of this equation, $\mathrm{S} = {}^{9}/_{13} = 0.6923$, a very low similarity, which WindowDiff also agrees upon ($1 - WD = 0.6154$). The edit-distance function $\mathrm{d}(s_{i1}, s_{i1}, T)$ can also be assigned values of the range $[0, 1]$ as scalar weights (w_{sub}, w_{trp}) to reduce the penalty attributed to particular edit operations, and configured to use a transposition error function (Equation 3, used by default).

3.4 Evaluating Automatic Segmenters

Coders often disagree in segmentation tasks (Hearst, 1997, p. 56), making it improbable that a single, correct, reference segmentation could be identified from human codings. This improbability is the result of individual coders adopting slightly different segmentation strategies (*i.e.*, different granularity). In light of this, we propose that the best available evaluation strategy for automatic segmentation methods is to compare performance against multiple coders directly, so that performance can be quantified relative to human reliability and agreement.

To evaluate whether an automatic segmenter performs on par with human performance, inter-annotator agreement can be calculated with and without the inclusion of an automatic segmenter, where an observed drop in the coefficients would signify that the automatic segmenter does not perform as reliably as the group of human coders.[7] This can be performed independently for multiple automatic segmenters to compare them to each other–assuming that the coefficients model chance agreement appropriately–because agreement is calculated (and quantifies reliability) over all segmentations.

3.5 Inter-Annotator Agreement

Similarity alone is not a sufficiently insightful measure of reliability, or agreement, between coders. Chance agreement occurs in segmentation when coders operating at slightly different granularities agree due to their codings, and not their own innate segmentation heuristics. Inter-annotator agreement coefficients have been developed that assume a variety of prior distributions to characterize chance agreement, and to attempt to offer a way to identify whether agreement is primarily due to chance, or not, and to quantify reliability.

Artstein and Poesio (2008) note that most of a coder's judgements are non-boundaries. The class imbalance caused by segmentations often containing few boundaries, paired with no handling of near misses, causes most inter-annotator agreement coefficients to drastically underestimate agreement on segmentations. To allow for agreement coefficients to account for near misses, we have adapted S for use with Cohen's κ, Scott's π, Fleiss's multi-π (π^*), and Fleiss's multi-κ (κ^*), which are all coefficients that range from $[{}^{A_e}/_{1-A_e}, 1]$, where 0 indicates chance agreement, and 1 perfect agreement. All four coefficients have the general form:

$$\kappa, \pi, \kappa^*, \text{ and } \pi^* = \frac{\mathrm{A}_a - \mathrm{A}_e}{1 - \mathrm{A}_e} \tag{5}$$

For each agreement coefficient, the set of categories is defined as solely the presence of a boundary ($K = \{\mathrm{seg}_t | t \in T\}$), per boundary type ($t$). This category choice is similar to those chosen by Hearst (1997, p. 53), who computed chance agreement in terms of the probability that coders would say that a segment boundary exists (seg_t), and the probability that they would not (unseg_t). We have chosen to model chance agreement only in terms of the presence of a boundary, and not the absence, because coders have only two choices when segmenting: to place a boundary, or not. Coders do not place non-boundaries. If they do not make a choice, then the default choice is used: no boundary. This default option makes it impossible to determine whether a segmenter is making a choice by not placing a boundary, or whether they are not sure whether a boundary is to be placed.[8] For this reason, we only characterize chance agreement between coders in terms of one boundary presence category per type.

with $\mathbf{t}$ boundary types is $\mathbf{t} \cdot \mathrm{mass}(i) - \mathbf{t}$.

[7] Similar to how human competitiveness is ascertained by Medelyan et al. (2009, pp. 1324–1325) and Medelyan (2009, pp. 143–145) by comparing drops in inter-indexer consistency.

[8] This could be modelled as another boundary type, which would be modelled in S by the set of boundary types T.

3.5.1 Scott's π

Proposed by Scott (1955), π assumes that chance agreement between coders can be characterized as the proportion of items that have been assigned to category k by both coders (Equation 7). We calculate agreement (A_a^π) as pairwise mean S (scaled by each item's size) to enable agreement to quantify near misses leniently, and chance agreement (A_e^π) can be calculated as in Artstein and Poesio (2008).

$$\mathrm{A}_a^\pi = \frac{\sum_{i \in I} \mathrm{mass}(i) \cdot \mathrm{S}(s_{i1}, s_{i2})}{\sum_{i \in I} \mathrm{mass}(i)} \tag{6}$$

$$\mathrm{A}_e^\pi = \sum_{k \in K} \left(\mathrm{P}_e^\pi(k)\right)^2 \tag{7}$$

We calculate chance agreement per category as the proportion of boundaries (seg_t) assigned by all coders over the total number of potential boundaries for segmentations, as shown in Equation 8.

$$\mathrm{P}_e^\pi(\mathrm{seg}_t) = \frac{\sum_{c \in C} \sum_{i \in I} |\mathrm{boundaries}(t, s_{ic})|}{\mathbf{c} \cdot \sum_{i \in I} \left(\mathrm{mass}(i) - 1\right)} \tag{8}$$

This adapted coefficient appropriately estimates chance agreement in situations where there no individual coder bias.

3.5.2 Cohen's κ

Proposed by Cohen (1960), κ characterizes chance agreement as individual distributions per coder, calculated as shown in Equations 9-10 using our definition of agreement (A_a^π) as shown earlier.

$$\mathrm{A}_a^\kappa = \mathrm{A}_a^\pi \tag{9}$$

$$\mathrm{A}_e^\kappa = \sum_{k \in K} \mathrm{P}_e^\kappa(k|c_1) \cdot \mathrm{P}_e^\kappa(k|c_2) \tag{10}$$

We calculate category probabilities as in Scott's π, but per coder, as shown in Equation 11.

$$\mathrm{P}_e^\kappa(\mathrm{seg}_t|c) = \frac{\sum_{i \in I} |\mathrm{boundaries}(t, s_{ic})|}{\sum_{i \in I} \left(\mathrm{mass}(i) - 1\right)} \tag{11}$$

This adapted coefficient appropriately estimates chance agreement for segmentation evaluations where coder bias is present.

3.5.3 Fleiss's Multi-π

Proposed by Fleiss (1971), multi-π (π^*) adapts Scott's π for multiple annotators. We use Artstein and Poesio's (2008, p. 564) proposal for calculating actual and expected agreement, and because all coders rate all items, we express agreement as pairwise mean S between all coders as shown in Equations 12-13, adapting only Equation 12.

$$\mathrm{A}_a^{\pi^*} = \frac{1}{\binom{\mathbf{c}}{2}} \sum_{m=1}^{\mathbf{c}-1} \sum_{n=m+1}^{\mathbf{c}} \frac{\sum_{i \in I} \mathrm{mass}(i) \cdot \mathrm{S}(s_{im}, s_{in})}{\sum_{i \in I} \left(\mathrm{mass}(i) - 1\right)} \tag{12}$$

$$\mathrm{A}_e^{\pi^*} = \sum_{k \in K} \left(\mathrm{P}_e^\pi(k)\right)^2 \tag{13}$$

3.5.4 Fleiss's Multi-κ

Proposed by Davies and Fleiss (1982), multi-κ (κ^*) adapts Cohen's κ for multiple annotators. We use Artstein and Poesio's (2008, extended version) proposal for calculating agreement just as in π^*, but with separate distributions per coder as shown in Equations 14-15.

$$\mathrm{A}_a^{\kappa^*} = \mathrm{A}_a^{\pi^*} \tag{14}$$

$$\mathrm{A}_e^{\kappa^*} = \sum_{k \in K} \left(\frac{1}{\binom{\mathbf{c}}{2}} \sum_{m=1}^{\mathbf{c}-1} \sum_{n=m+1}^{\mathbf{c}} \mathrm{P}_e^\kappa(k|c_m) \cdot \mathrm{P}_e^\kappa(k|c_n) \right) \tag{15}$$

3.6 Annotator Bias

To identify the degree of bias in a group of coders' segmentations, we can use a measure of variance proposed by Artstein and Poesio (2008, p. 572) that is quantified in terms of the difference between expected agreement when chance is assumed to vary between coders, and when it is assumed to not.

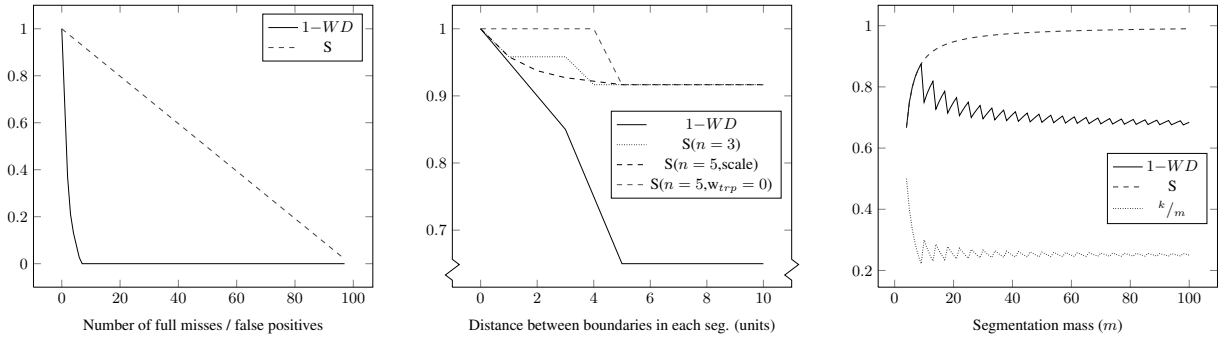
$$B = A_e^{\pi^*} - A_e^{\kappa^*} \tag{16}$$

4 Experiments

To demonstrate the advantages of using S, as opposed to WindowDiff (WD), we compare both metrics using a variety of contrived scenarios, and then compare our adapted agreement coefficients against pairwise mean WD[9] for the segmentations collected by Kazantseva and Szpakowicz (2012).

In this section, because WD is a penalty-based metric, it is reported as $1-WD$ so that it is easier to compare against S values. When reported in this way, $1-WD$ and S both range from $[0, 1]$, where 1 represents no errors and 0 represents maximal error.

[9] Permuted, and with window size recalculated for each pair.



(a) Increasing the number of full misses, or FPs, where $k = 25$ for WD

(b) Increasing the distance between two boundaries considered to be a near miss until metrics consider them a full miss

(c) Increasing the mass m of segmentations configured as shown in Figure 10 showing the effect of k on $1-WD$

Figure 5: Responses of $1-WD$ and S to various segmentation scenarios

4.1 Segmentation Cases

Maximal versus minimal segmentation When proposing a new metric, its reactions to extrema must be illustrated, for example when a maximal segmentation is compared to a minimal segmentation, as shown in Figure 6. In this scenario, both $1-WD$ and S appropriately identify that this case represents maximal error, or 0. Though not shown here, both metrics also report a similarity of 1.0 when identical segmentations are compared.

s_1	14													
s_2	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 6: Maximal versus minimal seg. masses

Full misses For the most serious source of error, full misses (*i.e.*, FPs and FNs), both metrics appropriately report a reduction in similarity for cases such as Figure 7 that is very similar ($1-WD = 0.8462$, S$= 0.8461$). Where the two metrics differ is when this type of error is increased.

s_1	1	2	2	2	4	2	1
s_2	1	2	8			2	1

Figure 7: Full misses in seg. masses

S reacts to increasing full misses linearly, whereas WindowDiff can prematurely report a maximal number of errors. Figure 5a demonstrates this effect, where for each iteration we have taken segmentations of 100 units of mass with one matching boundary at the first hypothesis boundary position, and uniformly increased the number of internal hypothesis segments, giving us 1 matching boundary, and $[0, 98]$ FPs. This premature report of maximal error (at 7 FP) by WD is caused by the window size ($k = 25$) being greater than all of the internal hypothesis segment sizes, making all windows penalized for containing errors.

Near misses When dealing with near misses, the values of both metrics drop ($1-WD = 0.8182$, S $= 0.9231$), but to greatly varying degrees. In comparison to full misses, WindowDiff penalizes a near miss, like that in Figure 8, far more than S. This difference is due to the distance between the two boundaries involved in a near miss; S shows, in this case, 1 PB of error until it is outside of the n-wise transposition window (where $n = 2$ PBs), at which point it is considered an error of not one transposition, but two substitutions (2 PBs).

s_1	6	8
s_2	7	7

Figure 8: Near misses in seg. masses

If we wanted to completely forgive near misses up to n PBs, we could set the weighting of transpositions in S to $w_{trp} = 0$. This is useful if a specific segmentation task accepts that near misses are very probable, and that there is little cost associated with a near miss in a window of n PBs. We can also set n to a high number, *i.e.*, 5 PBs, and use the scaled transposition error (te) function (Equation 3) to slowly increase the error from $b = 1$ PB to $b = 2$ PBs, as shown in Figure 5b, which shows how both

	Scenario 1: FN, $p = 0.5$		**Scenario 2: FP,** $p = 0.5$		**Scenario 3: FP and FN,** $p = 0.5$	
	(20,30)	(15,35)	(20,30)	(15,35)	(20,30)	(15,35)
WD	0.2340 ± 0.0113	0.2292 ± 0.0104	0.2265 ± 0.0114	0.2265 ± 0.0111	0.3635 ± 0.0126	0.3599 ± 0.0117
S	0.9801 ± 0.0006	0.9801 ± 0.0006	0.9800 ± 0.0006	0.9800 ± 0.0006	0.9605 ± 0.0009	0.9603 ± 0.0009
	(10,40)	(5,45)	(10,40)	(5,45)	(10,40)	(5,45)
WD	0.2297 ± 0.0105	0.2206 ± 0.0079	0.2256 ± 0.0102	0.2184 ± 0.0069	0.3516 ± 0.0110	0.3254 ± 0.0087
S	0.9799 ± 0.0007	0.9796 ± 0.0007	0.9800 ± 0.0006	0.9796 ± 0.0007	0.9606 ± 0.0010	0.9598 ± 0.0011

Table 1: Stability of mean (with standard deviation) values of WD and S in three different scenarios, each defining the: probability of a false positive (FP), false negative (FN), or both. Each scenario varies the range of internal segment sizes (*e.g.*, $(20, 30)$). Low standard deviation and similar within-scenario means demonstrates low sensitivity to variations in internal segment size.

metrics react to increases in the distance between a near miss in a segment of 25 units. These configurations are all preferable to the drop of $1-WD$.

4.2 Segmentation Mass Scale Effects

It is important for a segmentation evaluation metric to take into account the severity of an error in terms of segment size. An error in a 100 unit segment should be considered less severe than an error in a 2 unit segment, because an extra boundary placed within a 100 unit segment (*e.g.*, Figure 9 with $m = 100$) could probably indicate a weak boundary, whereas in a 4 unit segment the probability that an extra boundary exists right next to two agreed-upon boundaries should be small for most tasks, meaning that it is probable that the extra boundary is an error, and not a weak boundary.

s_1	$m/4$	$m/2$		$m/4$
s_2	$m/4$	$m/4$	$m/4$	$m/4$

Figure 9: Two segmentations of mass m with a full miss

To demonstrate that S is sensitive to segment size, Figure 5c shows how S and $1-WD$ respond when comparing segmentations configured as shown in Figure 10 (containing one match and one full miss) with linearly increasing mass ($4 \leq m \leq 100$). $1-WD$ will eventually indicate 0.68, whereas S appropriately discounts the error as mass is increased, approaching 1 as $\lim_{m\to\infty}$. $1-WD$ behaves in this way because of how it calculates its window size parameter, k, which is plotted as k/m to show how its value influences $1-WD$.

s_1	$m/4$	$m - (m/4)$	
s_2	$m/4$	$m/4$	$m/2$

Figure 10: Two segmentations of mass m compared with increasing m in Figure 5c (s_1 as reference)

4.3 Variation in Segment Sizes

When Pevzner and Hearst (2002) proposed WD, they demonstrated that it was not as sensitive as P_k to variations in the size of segments inside a segmentation. To show this, they simulated how WD performs upon a segmentation comprised of 1000 segments with four different uniformly distributed ranges of internal segment sizes (keeping the mean at approximately 25 units) in comparison to a hypothesis segmentation with errors (false positives, false negatives, and both) uniformly distributed within segments (Pevzner and Hearst, 2002, pp. 11–12). 10 trials were performed for each segment size range and error probability, with 100 hypotheses generated per trial. Recreating this simulation, we compare the stability of S in comparison to WD, as shown in Table 1. We can see that WD values show substantial within-scenario variation for each segment size range, and larger standard deviations, than S.

4.4 Inter-Annotator Agreement Coefficients

Here, we demonstrate the adapted inter-annotator agreement coefficients upon topical paragraph-level segmentations produced by 27 coders of 20 chapters from the novel *The Moonstone* by Wilkie Collins collected by Kazantseva and Szpakowicz (2012). Figure 11 shows a heat map of each chapter where the percentage of coders who agreed upon each potential boundary is represented. Comparing this heat map to the inter-annotator agreement coefficients in Table 2 allows us to better understand why certain chapters have lower reliability.

Chapter 1 has the lowest π^*_S score in the table, and also the highest bias (B_S). One of the reasons for this low reliability can be attributed to the chapter's small mass (m) and few coders ($|c|$), which makes it more sensitive to chance agreement. Visually, the

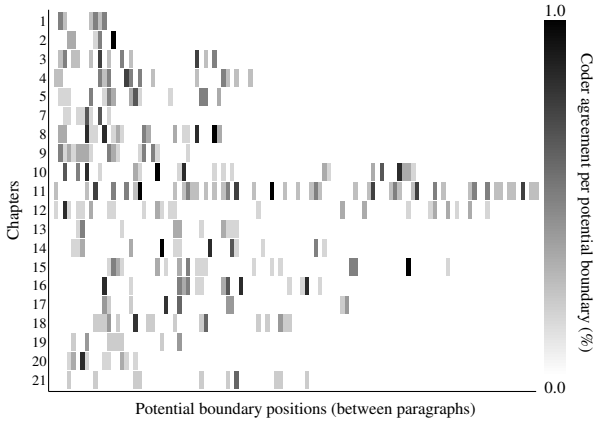


Figure 11: Heat maps for the segmentations of each chapter showing the percentage of coders who agree upon boundary positions (darker shows higher agreement)

Ch.	π^*_S	κ^*_S	B_S	$1{-}WD$	$\lvert c\rvert$	$\lvert b\rvert$	m
1	0.7452	0.7463	0.0039	0.6641 ± 0.1307	4	13	13
2	0.8839	0.8840	0.0009	0.7619 ± 0.1743	6	20	15
3	0.8338	0.8340	0.0013	0.6732 ± 0.1559	4	23	38
4	0.8414	0.8417	0.0019	0.6019 ± 0.2245	4	25	46
5	0.8773	0.8774	0.0003	0.6965 ± 0.1106	6	34	42
7	0.8132	0.8133	0.0002	0.6945 ± 0.1822	6	20	15
8	0.8495	0.8496	0.0006	0.7505 ± 0.0911	6	48	39
9	0.8104	0.8105	0.0009	0.6502 ± 0.1319	6	35	33
10	0.9077	0.9078	0.0002	0.7729 ± 0.0770	6	56	83
11	0.8130	0.8135	0.0022	0.6189 ± 0.1294	4	73	111
12	0.9178	0.9178	0.0001	0.6504 ± 0.1277	6	40	102
13	0.9354	0.9354	0.0002	0.5660 ± 0.2187	6	21	58
14	0.9367	0.9367	0.0001	0.7128 ± 0.1744	6	35	70
15	0.9344	0.9344	0.0001	0.7291 ± 0.0856	6	40	97
16	0.9356	0.9356	0.0000	0.8016 ± 0.0648	6	41	69
17	0.9447	0.9447	0.0002	0.6717 ± 0.2044	5	23	70
18	0.8921	0.8922	0.0005	0.5998 ± 0.1614	5	28	59
19	0.9021	0.9022	0.0009	0.4796 ± 0.2666	5	15	36
20	0.8590	0.8591	0.0003	0.6657 ± 0.1221	6	21	21
21	0.9286	0.9286	0.0004	0.6255 ± 0.2003	5	17	60

Table 2: S-based inter-annotator agreements and pairwise mean $1{-}WD$ and standard deviation with the number of coders, boundaries, and mass per chapter

predominance of grey indicates that, although there are probably two boundaries, their exact location is not very well agreed upon. In this case, $1{-}WD$ incorrectly indicates the opposite, that this chapter may have relatively moderate reliability, because it is not corrected for chance agreement.

$1{-}WD$ indicates that the lowest reliability is found in Chapter 19. π^*_S indicates that this is one of the higher agreement chapters, and looking at the heat map, we can see that it does not contain any strongly agreed upon boundaries. In this chapter, there is little opportunity to agree by chance due to the low number of boundaries ($|b|$) placed, and because the judgements are tightly clustered in a fair amount of mass, the S component of π^*_S appropriately takes into account the near misses observed and gives it a high reliability score.

Chapter 17 received the highest π^*_S in the table, which is another example of how tight clustering of boundary choices in a large mass leads π^*_S to appropriately indicate high reliability despite that there are not as many individual highly-agreed-upon boundaries, whereas $1{-}WD$ indicates that there is low reliability. $1{-}WD$ and π^*_S both agree, however, that chapter 16 has high reliability.

Despite WindowDiff's sensitivity to near misses, it is evident that its pairwise mean cannot be used to consistently judge inter-annotator agreement, or reliability. S demonstrates better versatility when accounting for near misses, and when used as part of inter-annotator agreement coefficients, it properly takes into account chance agreement. Following Artstein and Poesio's (2008, pp. 590–591) recommendation, and given the low bias (mean coder group $B_S = 0.0061 \pm 0.0035$), we propose reporting reliability using π^* for this corpus, where the mean coder group π^*_S for the corpus is 0.8904 ± 0.0392 (counting 1039 full and 212 near misses).

5 Conclusion and Future Work

We have proposed a segmentation evaluation metric which solves the key problems facing segmentation analysis today, including an inability to: appropriately quantify near misses when evaluating automatic segmenters and human performance; penalize errors equally (or, with configuration, in a manner that suits a specific segmentation task); compare an automatic segmenter directly against human performance; require a "true" reference; and handle multiple boundary types. Using S, task-specific evaluation of automatic and human segmenters can be performed using multiple human judgements unhindered by the quirks of window-based metrics.

In current and future work, we will show how S can be used to analyze hierarchical segmentations, and illustrate how to apply S to linear segmentations containing multiple boundary types.

Acknowledgments

We thank Anna Kazantseva for her invaluable feedback and corpora, and Stan Szpakowicz, Martin Scaiano, and James Cracknell for their feedback.

References

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596. MIT Press, Cambridge, MA, USA.

Doug Beeferman, Adam Berger, and John Lafferty. 1997. Text Segmentation Using Exponential Models. *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, 2:35–46. Association for Computational Linguistics, Stroudsburg, PA, USA.

Doug Beeferman and Adam Berger. 1999. Statistical models for text segmentation. *Machine learning*, 34(1–3):177–210. Springer Netherlands, NL.

Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46. Sage, Beverly Hills, CA, USA.

Frederick J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176. Association for Computing Machinery, Stroudsburg, PA, USA.

Mark Davies and Joseph L. Fleiss. 1982. Measuring agreement for multinomial data. *Biometrics*, 38(4):1047–1051. Blackwell Publishing Inc, Oxford, UK.

George R. Doddington. 1998. The topic detection and tracking phase 2 (TDT2) evaluation plan. *DARPA Broadcast News Transcription and Understanding Workshop*, pp. 223–229. Morgan Kaufmann, Waltham, MA, USA.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382. American Psychological Association, Washington, DC, USA.

Martin, Franz, J. Scott McCarley, and Jian-Ming Xu. 2007. User-oriented text segmentation evaluation measure. *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 701–702. Association for Computing Machinery, Stroudsburg, PA, USA.

William Gale, Kenneth Ward Church, and David Yarowsky. 1992. Estimating upper and lower bounds on the performance of word-sense disambiguation programs. *Proceedings of the 30th annual meeting of the Association for Computational Linguistics*, pp. 249–256. Association for Computational Linguistics, Stroudsburg, PA, USA.

Maria Georgescul, Alexander Clark, and Susan Armstrong. 2006. An analysis of quantitative aspects in the evaluation of thematic segmentation algorithms. *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, pp. 144–151. Association for Computational Linguistics, Stroudsburg, PA, USA.

Marti A. Hearst. 1997. TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages. *Computational Linguistics*, 23(1):33–64. MIT Press, Cambridge, MA, USA.

Anna Kazantseva and Stan Szpakowicz. 2012. Topical Segmentation: a Study of Human Performance. *Proceedings of Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA.

Klaus Krippendorff. 1980. *Content Analysis: An Introduction to Its Methodology*, Chapter 12. Sage, Beverly Hills, CA, USA.

Klaus Krippendorff. 2004. *Content Analysis: An Introduction to Its Methodology*, Chapter 11. Sage, Beverly Hills, CA, USA.

Sylvain Lamprier, Tassadit Amghar, Bernard Levrat, and Frederic Saubion 2007. On evaluation methodologies for text segmentation algorithms. *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence*, 2:19–26. IEEE Computer Society, Washington, DC, USA.

Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710. American Institute of Physics, College Park, MD, USA.

Olena Medelyan. 2009. Human-competitive automatic topic indexing. PhD Thesis. University of Waikato, Waikato, NZ.

Olena Medelyan, Eibe Frank, and Ian H. Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pp. 1318–1327. Association for Computational Linguistics, Stroudsburg, PA, USA.

Rebecca J. Passonneau and Diane J. Litman. 1993. Intention-based segmentation: human reliability and correlation with linguistic cues. *Proceedings of the 31st annual meeting of the Association for Computational Linguistics*, pp. 148–155). Association for Computational Linguistics, Stroudsburg, PA, USA.

Lev Pevzner and Marti A. Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1):19–36. MIT Press, Cambridge, MA, USA.

William A. Scott. 1955. Reliability of content analysis: The case of nominal scale coding. *Public Opinion Quarterly*, 19(3):321–325. American Association for Public Opinion Research, Deerfield, IL, USA.

Sidney Siegel and N. John Castellan, Jr. 1988. *Nonparametric Statistics for the Behavioral Sciences*. 2nd Edition, Chapter 9.8. McGraw-Hill, New York, USA.