Sentence Boundary Detection: A Long Solved Problem?

Jonathon READ, Rebecca DRIDAN, Stephan OEPEN, Lars Jørgen SOLBERG
University of Oslo, Department of Informatics
{jread|rdridan|oe|larsjol}@ifi.uio.no

ABSTRACT
We review the state of the art in automated sentence boundary detection (SBD) for English and call for a renewed research interest in this foundational first step in natural language processing. We observe severe limitations in comparability and reproducibility of earlier work and a general lack of knowledge about genre- and domain-specific variations. To overcome these barriers, we conduct a systematic empirical survey of a large number of extant approaches, across a broad range of diverse corpora. We further observe that much previous work interpreted the SBD task too narrowly, leading to overly optimistic estimates of SBD performance on running text. To better relate SBD to practical NLP use cases, we thus propose a generalized definition of the task, eliminating text- or language-specific assumptions about candidate boundary points. More specifically, we quantify degrees of variation across 'standard' corpora of edited, relatively formal language, as well as performance degradation when moving to less formal language, viz. various samples of user-generated Web content. For these latter types of text, we demonstrate how moderate interpretation of document structure (as is now often available more or less explicitly through mark-up) can substantially contribute to overall SBD performance.

KEYWORDS: Sentence Boundary Detection, Segmentation, Comparability, Reproducibility.

Proceedings of COLING 2012: Posters, pages 985–994,
COLING 2012, Mumbai, December 2012.

1 Motivation and Introduction

Sentence Boundary Detection (SBD) is not widely counted among the grand challenges in NLP. Even though there were comparatively few studies on SBD in the past decades, the assessment of extant techniques for English is hindered by variation in the task definition, choice of evaluation metrics, and test data used. Furthermore, two development trends in NLP pose new challenges for SBD, viz. (a) a shift of emphasis from formal, edited text towards more spontaneous language samples, e.g. Web content; and (b) a gradual move from 'bare' ASCII to *rich* text, exploiting the much wider Unicode character range as well as mark-up of text structure. The impact of such textual variation on SBD is hardly explored, and off-the-shelf technologies may perform poorly on text that is not very newswire-like, i.e. different from the venerable Wall Street Journal (WSJ) collection of the Penn Treebank (PTB; Marcus et al., 1993).

In this work, we seek to provide a comprehensive, up-to-date, and fully reproducible assessment of the state of the art in SBD. In much NLP research, the 'sentence' (in a suitable interpretation; see below) is a *foundational* unit, for example in aligning parallel texts; PoS tagging; syntactic, semantic, and discourse parsing; or machine translation. Assuming gold-standard sentence boundaries (and possibly tokenisation)—as provided by standard data sets like the PTB—has been common practice for many isolated studies. However, strong effects of error propagation must be expected in standard NLP pipelines, for example of imperfect SBD into morpho-syntactic, semantic, or discourse analysis (Walker et al., 2001; Kiss and Strunk, 2002). For these reasons, we aim to determine (a) what levels of performance can be expected from extant SBD techniques; (b) to which degree SBD performance is sensitive to variation in text types; and (c) whether there are relevant differences in observed behavior across different SBD approaches.

Our own motivation in this work is twofold: First, working in the context of semi-automated parser adaptation to domain and genre variation, we would hope to encourage a shift of emphasis towards parsing as an end-to-end task, i.e. taking as its point of departure the running text of a document collection rather than idealized resources comprised of 'pure' text with manually annotated, gold-standard sentence and token boundaries. Second, in preparing new annotated language resources (encompassing a broader range of different text types), we wish to identify and adapt extant preprocessing tool(s) that are best suited to our specific needs—both to minimize the need for correction in manual annotation and to maximize quality in automatically produced annotations. Finally, we felt prompted into systematizing this work by a recent query (for SBD technology, a recurrent topic) to the CORPORA mailing list[1], where a wealth of subjective recommendations were offered, but very few 'hard' empirical results.

This article first offers a concise summary of previous SBD work (§ 2) and data sets and evaluation metrics applicable to SBD (§ 3).[2] For nine publicly available SBD technologies and against four 'standard' data sets, § 4 surveys the current state of the art. Next, § 5 turns to trends in 'Web-scale' NLP sketched earlier, viz. processing more informal, more varied language: here, we extend the experimental SBD survey with results on a variety of samples of user-generated content and quantify the potential contribution of mark-up analysis to SBD. Throughout the text, we use the term *sentence* in a non-syntactic sense, viz. to refer to all types of root-level utterances (including, for example, noun or verb phrases), i.e. text units whose relations to surrounding context are not primarily of a syntactic but rather of a rhetorical, discourse-level nature. Nunberg (1990) coins the term *text-sentence* for this unit, and his definition seems to capture well the variation of segment types we find in our various gold-standard SBD data sets.

[1] See `http://listserv.linguistlist.org/cgi-bin/wa?A2=CORPORA;545fc67c.1208`.

[2] For comparability in future work, all our resources are available at `http://svn.delph-in.net/odc/`.

2 Previous Work on Sentence Segmentation

Approaches to sentence segmentation broadly fall into three classes: (a) *rule-based* SBD, using hand-crafted heuristics and lists (of abbreviations, for example); (b) machine learning approaches to SBD trained in a *supervised* setup, i.e. on text annotated with gold-standard boundaries; and (c) *unsupervised* applications of machine learning, requiring only raw, unannotated corpora. There is comparatively little SBD research literature, and with some notable exceptions, published studies focus on machine learning approaches. Conversely, as we see in § 4 below, many existing tools actually rely on heuristic approaches, but these have very rarely been compared empirically to each other or to the performance of machine learning techniques. As such, despite an imbalance in available literature, the choice of SBD approach and relevant trade-offs—e.g. in terms of maximum accuracy vs. robustness to variations in text types—have remained open questions, both methodologically and technologically.

Riley (1989) presents an early application of machine learning to SBD, investigating the use of decision tree classifiers in determining whether instances of full stops (periods, in American English) mark sentence boundaries. As we argue in § 3 below, limiting SBD to the disambiguation of a single punctuation mark (or a small set of such) is an overly narrow interpretation of the task—even for formal, edited language—albeit quite characteristic for the vast majority of SBD research reports. The approach of Riley (1989) utilizes features including the probabilities of words being sentence-final or -initial, word length, and word case. When training on 25 million words of AP newswire and testing on the Brown Corpus (Francis and Kucera, 1982), they report an accuracy of 99.8%.

The `SATZ` system (Palmer and Hearst, 1997) performs sentence segmentation using models of the part-of-speech distribution of the context surrounding a potential sentence boundary. The basis of the system involves representing contextual words as vectors of binary values with elements that indicate whether or not a part-of-speech tag is plausible for that word. Evaluating their system against WSJ text, they report an error rate of 1.1% using a neural net, and 1.0% using a decision tree. Creating a hybrid system by integrating their classifier with a heuristics-based approach, reduces the error rate to 0.5%.

Reynar and Ratnaparkhi (1997) employ supervised Maximum Entropy learning. Both their system variants treat segmentation as a disambiguation task, wherein every token containing '!', '.' or '?' is a potential sentence boundary. The first system aims for premium in-domain performance, using features targeting English financial use (e.g. identification of honorifics and corporate designations). The second system is designed to be more portable and uses domain-independent features, including a list of abbreviations derived from the training data. Testing on both WSJ text and the Brown Corpus, they report accuracies of 98.8% and 97.9%, respectively, for the domain-dependent system, and of 98.0% and 97.5%, respectively, for the portable system. More recently, Gillick (2009) discusses further feature development for supervised classification, but concentrates on the narrower problem of full stops as candidate boundaries. The best configuration of his system, `Splitta`, employs Support Vector Machines as a learning framework, with reported error rates of 0.25% on WSJ and 0.36% on Brown.

Mikheev (2002) treats sentence segmentation with a small set of rules based on determining whether the words to the left or right of a potential sentence boundary are abbreviations or proper names (which are derived from heuristics on unlabeled training data). Mikheev (2002) reports an error rate of 0.45% on WSJ text and 0.28% on the Brown Corpus. Combining this approach with a supervised part-of-speech tagger that includes tags for end of sentence markers (Mikheev, 2000) further reduces the error rates to 0.31% and 0.20%, respectively.

A fully unsupervised system, `Punkt`, is presented by Kiss and Strunk (2006). The approach is rooted in the identification of abbreviations, by finding collocational bonds between candidates and full stops. Their reported rate of errors on 'classic' test sets is 1.02% (Brown) and 1.65% (WSJ), but further experiments on other data sets demonstrate the system's usefulness across a broad variety of text types and languages.

3 Reflections: SBD Resources and Metrics

Comparability and reproducibility are lacking in previous work, due to divergent interpretations of the SBD task and often vaguely specified test sets and evaluation metrics. Different assumptions about which and how many candidate boundary points to consider, for example, directly impact the relative difficulty of the task; likewise, although much previous work involved one or both of WSJ and Brown texts, historically there have been different (more or less) annotated versions of these resources, and several published studies indicate that preparation of SBD test data from these sources involved some amount of manual 'correction'. In order to conduct a more systematic survey, we assembled various publicly available data sets where both raw text and gold-standard sentence segmentation were possible to download or reconstruct. We also suggest a formal, generalized definition of the SBD task that we feel better represents the practical problem, from the point of view of an NLP pipeline.

Gold-Standard Data Sets Our aim is to produce data sets that are as faithful as possible to the original text form, including paragraph breaks. For some continuity with previous work, we use both WSJ and Brown Corpus data, which required some corpus archaeology and reconstruction. An alignment between the original text of the WSJ (last released in 1995 as LDC#95T07) and the annotations in the PTB has recently been published (Dridan and Oepen, 2012), and we thus start from raw, untokenised text and superimpose onto it 7,706 PTB gold-standard sentence boundaries (from Sections 3–6, the 'classic' SBD subset). Various versions of the Brown Corpus exist, but none correspond exactly to the original raw text. For our SBD tests, we combine sentence segmentation in the tagged Brown Corpus (distributed with the Natural Language Toolkit, NLTK) with the 'raw', so-called *Bergen Form I*[3] as a reference for automatically reversing tokenisation, quote disambiguation, and some artifacts that appear only in the tagged version. As in previous work, we also occasionally corrected the segmentation (often related to paragraph-initial quote marks, which in the tagged data can occur as a 'sentence' by themselves), and restored punctuation to better match the original raw text—for a total of 57,275 sentences.

We also used more recently released data, to explore a wider range of edited text types and introduce 'fresh' test data. The Conan Doyle Corpus (CDC) was used in the 2012 *SEM Shared Task (Morante and Blanco, 2012), and provides various Sherlock Holmes stories in both raw and segmented versions, albeit for only 5,692 sentences. Then, from quite a different domain, the GENIA Corpus, a collection of 16,392 sentences from biomedical research abstracts (Kim et al., 2003), also contains the required boundary annotation. The use of these additional corpora allows us to assess the generality of the various tools, which is particularly important as WSJ and Brown data has been central in much previous SBD development.

Task Definition and Evaluation In much previous work, SBD was operationalized as a binary classification of a fixed number of candidate boundary points—restricted to, for example, full stops, token-final full stops, or a small set of sentence-terminating punctuation. While this makes for a clean machine learning task, we consider that it over-simplifies the SBD problem, specifically in not directly penalizing for missing gold-standard boundary points that do not fall

[3] As described at `http://icame.uib.no/brown/bcm.html`.

CoreNLP	R	18.7	http://nlp.stanford.edu/software/corenlp.shtml
GATE	R	60.6	http://gate.ac.uk
LingPipe	R	1.7	http://alias-i.com/lingpipe/
MxTerminator	S	2.8	ftp://ftp.cis.upenn.edu/pub/adwait/jmx/
OpenNLP	S	2.2	http://opennlp.apache.org/
Punkt	U	7.1	http://nltk.org/api/nltk.tokenize.html
RASP	R	0.3	http://ilexir.co.uk/applications/rasp/
Splitta	S	31.5	http://code.google.com/p/splitta
tokenizer	R	0.3	http://www.cis.uni-muenchen.de/~wastl/misc/

Table 1: Overview of publicly available SBD systems in our survey. The table columns indicate, from left to right, the general approach (rule-based, supervised, or unsupervised); total (wall-clock) number of seconds to segment the Brown Corpus on an unloaded workstation; and download site.

onto a classification candidate. Even on the edited 'standard' texts in our survey, the percentage of sentences ending in a full stop is only 87.7 (91.9% for sentence-final '.', '?', or '!'). We anticipate that this percentage will drop further in less formal texts. Thus, to give a more representative picture of how the sentence segmenters succeed in segmenting raw running text, we propose a more general definition of the task, which considers the positions after *every* character as a potential boundary point. Hence, any gold-standard boundary missed will be counted as a false negative, even if there was no punctuation mark at that point. This makes the set of candidate boundaries not only much larger, but also very heavily weighted towards the negative, uninteresting class. As such, we evaluate SBD in terms of precision, recall, and F_1 over boundary points, since accuracy or error rate (used in much of the previous work) would be both less informative and more difficult to compare for the task as defined here.

4 Surveying the State of the Art

Our survey covers nine publicly-available SBD systems and toolkits with sentence segmentation components, as summarized in Table 1.[4] Although purely technical aspects (e.g. supported platforms or available APIs) are not in focus here, we seek to provide a coarse indication of run-time efficiency in terms of total processing time when segmenting the Brown Corpus as one single document. For high-throughput use cases, the comparatively 'lean', heuristic systems `RASP` and `tokenizer`—building on the Un*x `(f)lex` tool in the tradition of Grefenstette and Tapanainen (1994)—may have an advantage. Note that one tool offers two pre-packaged configurations: one for 'general' text, here dubbed $\texttt{LingPipe}_1$, and another tuned to biomedical literature and specifically the GENIA Corpus, $\texttt{LingPipe}_2$. We were unable to obtain implementations for the earlier studies of Palmer and Hearst (1997) and Mikheev (2002).

Table 2 presents SBD performance levels for our nine systems and four corpora in a first, off-the-shelf experiment. Here, we sought to run each tool in its default configuration—often instantiating sample invocations or API calls, where available—and in a setup we consider representative for an NLP pipeline: processing each corpus as a contiguous text stream. For `Punkt`, we rely on the implementation and pre-trained model for English that ships with NLTK (Bird et al., 2009). Recall from § 3 above that our corpus preparation preserves in-text paragraph boundaries (as indicated in plain raw text by consecutive double line breaks); where a corpus internally is comprised of multiple segments (e.g. the distinct sections of the PTB or separate stories in CDC), we inserted paragraph breaks between segments.

In this first experiment, we see stark variation between tools and across corpora. We were

[4]The table also includes a tenth system, the `GATE` text processing environment. However, we have not been able to empirically determine its SBD performance, as `GATE` (at least in its standard configuration) often fails to reproduce the full input in its output (for example dropping parts of date expressions or sentence-initial quote marks).

	Brown	CDC	GENIA	WSJ	All
CoreNLP	87.7	72.1	98.8	91.3	89.1
LingPipe$_1$	94.9	87.6	98.3	97.3	95.2
LingPipe$_2$	93.0	86.3	**99.6**	88.0	93.2
MxTerminator	94.7	97.9	98.3	97.4	95.8
OpenNLP	96.6	**98.6**	98.8	99.1	**97.4**
Punkt	96.4	98.7	99.3	98.3	97.3
RASP	**96.8**	96.1	98.9	99.0	**97.4**
Splitta	95.4	96.1	99.0	**99.2**	96.5
tokenizer	94.9	**98.6**	98.6	97.9	96.2

Table 2: Out-of-the-box SBD performance (F_1 over sentence boundary points) over a sample of edited English corpora. Note that many of the tools were trained on or tuned towards variants of WSJ, Brown, or GENIA, and hence some of the scores may overestimate general performance on these text types.

surprised at comparatively low performance levels for some of the tools (e.g. CoreNLP and MxTerminator), and further note that some systems seem far less robust to corpus variation than others, in particular showing relatively drastic performance drops on CDC (e.g. CoreNLP, LingPipe, RASP, or Splitta). To investigate this variation, we performed a coarse error analysis and found two main issues: First, not all SBD tools interpret paragraph boundaries, whereas forcing sentence boundaries there would seem a strong and practical heuristic rule. Second, sentence-final punctuation can of course be followed by closing quotation marks,[5] and it appears that the 'modern' Unicode quotes in CDC are not recognized by all tools.

To control for these factors, we ran a second batch of experiments and allowed what we consider a reasonable amount of preprocessing of input texts, to better match tool-internal assumptions: (a) slicing our corpora into separate 'documents' at paragraph boundaries, i.e. effectively forcing sentence boundaries there; (b) substituting Unicode directional quote marks ⟨‘ ’ “ ”⟩ with straight ASCII quotes ⟨' "⟩ or (c) substituting Unicode quotes with the directional variants used in LaTeX and the PTB ⟨` ' `` ''⟩.[6] Table 3 shows resulting SBD performance, where for each system we ran an additional four configurations—with or without paragraph slicing, and applying either scheme of quote substitution—and report the best-performing setup.

The results in Table 3 no longer show surprising outliers and, in our view, offer better indicators of available performance levels across text types. Only in very few cases do we observe performance levels as suggested by some previous reports (e.g. 99.8 for LingPipe$_2$ on GENIA and 99.1 and 99.2 for OpenNLP and Splitta, respectively, on WSJ)—and these are very likely owed to training or tuning against these very data sets. In this regard, CDC may be our most independent indicator of SBD performance, although we note that due to its fictional nature it is especially rich in quoted speech. The unsupervised system in this survey, Punkt, appears reasonably competitive on Brown, CDC, and GENIA and achieves an overall fifth rank (outperforming CoreNLP, LingPipe$_2$, MxTerminator, and Splitta). In our view, these results bode well for adaptability to new text types or languages. At the same time, comparatively poor performance of Punkt on WSJ text may be a key reason for its 'not quite state-of-the-art' reputation. In future work, it would be interesting to break down corpus properties that affect SBD performance and, thus, seek to identify which WSJ characteristics prove especially challenging for the unsupervised machine-learning approach.

[5] This is the case for question marks and exclamation points in both prevalent typographic conventions for quotations, which are at times called American- and British-style. Further, full stops (and also commas, though these are not typically sentence boundary cues) can precede a closing quotation mark in American-style typography.

[6] We note that either quote substitution scheme may introduce low-level technical hurdles into the NLP pipeline, as scheme (b) loses the distinction between opening and closing quote marks, while scheme (c) substitutes two characters for one, which may complicate book-keeping of character points against the original document.

	Brown	CDC	GENIA	WSJ	All
CoreNLP	$^{+}$93.6	$^{L+}$98.3	$^{+}$99.0	$^{+}$94.8	95.0
LingPipe$_1$	$^{+}$96.6	$^{A+}$99.1	$^{+}$98.6	$^{+}$98.7	97.4
LingPipe$_2$	$^{+}$94.5	$^{+}$97.2	$^{+}$**99.8**	$^{+}$90.9	95.3
MxTerminator	$^{+}$96.5	$^{+}$98.6	$^{+}$98.5	$^{+}$98.5	97.2
OpenNLP	96.6	98.6	98.8	99.1	97.4
Punkt	96.4	98.7	99.3	98.3	97.3
RASP	96.8	^{A}99.1	98.9	99.0	**97.6**
Splitta	95.4	^{A}96.7	99.0	**99.2**	96.5
tokenizer	$^{+}$**96.9**	$^{+}$**99.2**	$^{+}$98.9	$^{+}$**99.2**	**97.6**

Table 3: Best-case SBD performance. Results prefixed with '+' indicate forcing sentence boundaries at paragraph breaks, while the 'A' and 'L' prefixes mark substitution of Unicode quotes to ASCII or LaTeX-style, respectively. Non-prefixed scores are repeated from Table 2 for ease of comparison.

Among the supervised machine-learning tools, MxTerminator and OpenNLP show very similar results, confirming OpenNLP as essentially a reimplementation of Reynar and Ratnaparkhi (1997), if maybe with a slight edge over the original. Splitta, representing the most recent SBD study applying supervised learning, on the other hand, is outranked by OpenNLP by almost one full F_1 point. It further shows comparatively larger drops on Brown and CDC, which taken together with its premium performance on WSJ could be suggestive of limited robustness to text variation (or over-tuning effects). Three of the rule-based tools, in this survey, show comparable behavior: LingPipe$_1$ (the 'generic' variant), RASP, and tokenizer all deliver relatively good results across the board and show comparatively limited sensitivity to variation in text types. Keeping in mind the caveat of being developed against the exact same type of texts, LingPipe$_2$ on GENIA confirms the potential benefits from SBD adaptation to a specific genre and domain. Finally, CoreNLP and LingPipe differ in architecture from the other rule-based systems, in that they apply SBD to a stream of tokens rather than a raw text stream. Hence, overall lower performance in CoreNLP could in principle be an effect of error propagation from the preceding tokenisation phase (Dridan and Oepen, 2012), or may just be owed to this (part of a larger) system not being as thoroughly engineered as the specialized RASP or tokenizer rule sets.

5 SBD for More Informal, User-Generated Content

To extend our survey to rather different types of text—user-generated Web content (UGC)—we ran similar experiments on two recent corpora that come with gold-standard sentence boundary annotations: First, the WeScience Corpus of Ytrestøl et al. (2009) comprises some 12,000 sentences from Wikipedia, drawing on a sample of articles in the NLP domain. Second, the WeSearch Data Collection (WDC; Read et al., 2012) includes gold-standard annotations for two smaller samples of Web blogs, some 1,000 sentences each in the NLP and Linux domains, respectively (dubbed WNB and WLB in Table 4 below). As none of our SBD tools fully supports mark-up processing, we reduced the WeScience and WDC corpora into a pure text form, only keeping paragraph breaks from the original mark-up in a first experimental setup dubbed *A* in Table 4. A variant experiment, dubbed *B* below, aims to gauge the potential contribution of mark-up to SBD, where we insert additional paragraph boundaries around block elements like headings, pre-formatted text, and individual elements of lists.

Comparing variants *A* and *B* in Table 4, there is no doubt that forcing sentence boundaries around select mark-up elements much improves SBD on the two blog fragments, even if our WNB and WLB scores draw on comparatively small test sets. Therefore, we focus on setup *B* in the subsequent discussion. Intuitively, there is a decline in linguistic formality along the horizontal dimension of Table 4: in spite of greater author diversity in Wikipedia, personal blogs are likely less thoroughly edited and possibly more creative in their language use; furthermore,

	WeScience		WNB		WLB	
	A	***B***	***A***	***B***	***A***	***B***
`CoreNLP`$^{+L}$	90.0	97.9	95.3	96.4	89.1	90.9
`LingPipe`$_1^{+A}$	90.0	98.1	94.8	96.1	92.4	94.2
`LingPipe`$_2^{+}$	89.8	98.0	94.4	95.6	92.7	94.5
`MxTerminator`$^{+}$	89.5	97.2	94.7	95.9	90.3	92.2
`OpenNLP`	90.2	97.9	95.3	96.5	90.2	92.0
`Punkt`	89.9	97.7	**95.6**	96.7	92.8	94.5
`RASP`^{A}	**91.0**	99.1	95.4	96.6	92.8	94.6
`Splitta`^{A}	**91.0**	98.9	94.0	95.5	91.2	93.4
`tokenizer`$^{+}$	**91.0**	**99.2**	**95.6**	**96.8**	**93.1**	**94.9**

Table 4: SBD performance over a sample of user-generated English content. Paragraph slicing and quote substitution were applied as per the individual best-case configurations on CDC, as indicated in Table 3.

bloggers in the Linux domain may care less about linguistic form than NLP bloggers, and quite possibly make more use of technical 'slang'. Average SBD performance levels appear to match these intuitions, where top Wikipedia scores in fact are higher than the best average F_1 over our 'formal' corpora in Table 3 above. We conjecture that this difference is probably owed to our mark-up processing and a relatively high proportion of headings and list elements.

While average performance levels on WNB and especially WLB are markedly lower, the magnitudes of differences in system scores seem somewhat reduced in all UGC experiments. However, we see several earlier observations confirmed, with `tokenizer` first and `RASP` (in total) a close second—consistently across data sets (despite a potential bias in favour of `tokenizer` on WeScience due to its use in the construction of that corpus). Relative ranks of most other tools vary somewhat with the data sets, suggesting variable robustness to pertinent stylistic elements. Still, `OpenNLP` (on average) improves mildly over `MxTerminator`, whereas among the rule-based systems `LingPipe`$_1$ now often patterns more with `CoreNLP` than with the top performers. Maybe most notably, the unsupervised `Punkt` performs close to the top scores for WDC. It would be interesting to extend the survey further towards more 'noisy' Web text.

6 Conclusions—Future Work

In this work, we have sought to compile a comprehensive and fair assessment of the state of the art in sentence boundary detection—both for our own benefit and in the hope that this survey may be of wider interest. To better relate to practical use cases of SBD as a foundational element in the NLP pipeline, we generalize the task definition to recovering *all* gold-standard sentence boundaries in *running* text. For this more realistic definition of the task, performance levels upwards of 99% (as previously reported) are not generally available. Our results establish, for the first time, comparability and reproducibility across a broad range of approaches and text types, including novel test corpora and a systematic exploration of performance degradation along the dimension of linguistic formality. We anticipate possible further calibration in dialogue with tool developers and plan on publishing our data sets and results as part of the *State of the Art* Section on the ACL Wiki. In doing so, we hope to stimulate new research in SBD, particularly aiming to improve performance on 'noisy' language, increase robustness to domain and genre variation, and take better advantage of rich text properties and structure.

In ongoing work, we aim to combine some of the attractions in unsupervised learning with the robustness of heuristic rules, for example extending a tool like `tokenizer` with automatically acquired and domain-adapted lists of abbreviations. Furthermore, we believe that tighter integration of mark-up processing and SBD is a prerequisite to better results on user-generated Web content. Some of the tools in our survey (notably `GATE`) provide some HTML support, and we hope to assess the efficacy of mark-up modes, where available, in the near future.

References

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly, Beijing.

Dridan, R. and Oepen, S. (2012). Tokenization. Returning to a long solved problem. A survey, contrastive experiment, recommendations, and toolkit. In *Proceedings of the 50th Meeting of the Association for Computational Linguistics*, page 378–382, Jeju, Republic of Korea.

Francis, W. N. and Kucera, H. (1982). *Frequency Analysis of English Usage*. Houghton Mifflin Co., New York.

Gillick, D. (2009). Sentence boundary detection and the problem with the U.S. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, page 241–244, Boulder, CO, USA.

Grefenstette, G. and Tapanainen, P. (1994). What is a word, what is a sentence? Problems of tokenization. In *Proceedings of the 3rd Conference on Computational Lexicography and Text Research*, page 79–87, Budapest, Hungary.

Kim, J.-D., Ohta, T., Teteisi, Y., and Tsujii, J. (2003). GENIA corpus — a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19:i180–i182.

Kiss, T. and Strunk, J. (2002). Viewing sentence boundary detection as collocation identification. In *Proceedings of 6. Konferenz zur Varabeitung natürlicher Sprache*, page 75–82, Saarbrücken, Germany.

Kiss, T. and Strunk, J. (2006). Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525.

Marcus, M., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpora of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.

Mikheev, A. (2000). Tagging sentence boundaries. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, page 264–271, Seattle, WA, USA.

Mikheev, A. (2002). Periods, capitalized words, etc. *Computational Linguistics*, 28(3):289–318.

Morante, R. and Blanco, E. (2012). *SEM 2012 shared task: Resolving the scope and focus of negation. In *Proceedings of the 1st Joint Conference on Lexical and Computational Semantics*, page 265–274, Montréal, Canada.

Nunberg, G. (1990). *The Linguistics of Punctuation*. Number 18 in Lecture Notes. CSLI Publications, Stanford, CA.

Palmer, D. D. and Hearst, M. A. (1997). Adaptive multilingual sentence boundary disambiguation. *Computational Linguistics*, 23(2):242–267.

Read, J., Flickinger, D., Dridan, R., Oepen, S., and Øvrelid, L. (2012). The WeSearch Corpus, Treebank, and Treecache. A comprehensive sample of user-generated content. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, Istanbul, Turkey.

Reynar, J. C. and Ratnaparkhi, A. (1997). A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, page 16 – 19, Washington, D.C., USA.

Riley, M. D. (1989). Some applications of tree-based modelling to speech and language. In *Proceedings of the DARPA Speech and Natural Language Workshop*, page 339 – 352.

Walker, D. J., Clements, D. E., Darwin, M., and Amtrup, J. W. (2001). Sentence boundary detection: A comparison of paradigms for improving MT quality. In *Proceedings of the MT Summit VIII*, Santiago de Compostela, Spain.

Ytrestøl, G., Oepen, S., and Flickinger, D. (2009). Extracting and annotating Wikipedia subdomains. In *Proceedings of the 7th International Workshop on Treebanks and Linguistic Theories*, page 185 – 197, Groningen, The Netherlands.