Viewing sentence boundary detection as collocation identification

Tibor Kiss
Sprachwissenschaftliches Institut
Ruhr-Universität Bochum
D-44780 Bochum
tibor@linguistics.ruhr-uni-bochum.de

Jan Strunk
Sprachwissenschaftliches Institut
Ruhr-Universität Bochum
D-44780 Bochum
strunk@linguistics.ruhr-uni-bochum.de

Abstract

The detection of abbreviations is an important step in the process of sentence boundary detection. We describe a flexible, language-independent and accurate method based on the idea that an abbreviation can be viewed as a collocation. As such, it can be identified by using methods for collocation detection such as the *log likelihood ratio.* Although the log likelihood ratio is known to show a good recall, its precision is poor. We employ scaling factors that lead to a strong improvement of precision. Experiments with English and German corpora show that abbreviations can be detected with high accuracy. We also show that inaccurate tokenization leads to a considerably higher error rate during tagging.

Introduction

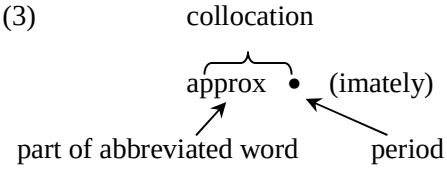
The detection of abbreviations in a text corpus forms one of the initial steps in tokenization (cf. Liberman/Church 1992). This is not a trivial task, since a tokenizer is confronted with ambiguous tokens. For English, e.g., Palmer/Hearst (1997:241) report that *periods* (•) can be used as decimal points, abbreviation marks, end-of-sentence marks, and as abbreviation marks at the end of a sentence, cf. (1). Moreover, the usage of the period differs from language to language. As can be seen in (2), the period is additionally used after ordinal numbers in German.

(1) CELLULAR COMMUNICATIONS INC. sold 1,550,000 common shares at $21.75 each yesterday, according to lead underwriter L.F. Rothschild & Co.

(2) «Der Artikel ‹Zwielichtige Fürsprecher für das Tier› in der NZZ vom 27. Januar 1993 (Nr. 21) verweist auf eine Reportage des ‹Beobachters› vom 25. März 1988 betreffend die Tierhaltung von Frau Dr. Milly Schär-Manzoli, welche Falschbehauptungen enthält.

In this paper, we will concentrate on the classification of the period as either an abbreviation mark or a punctuation mark. The period is generally considered to form part of the token in abbreviations and should thus be left attached. It should be separated as an individual token in case of an end-of-sentence mark.[1] We assume that an abbreviation can be viewed as a *collocation* consisting of the abbreviated word itself and the following •.

(3)



In case of an abbreviation, we expect the occurrence of • following the previous ‘word’ to be more likely than in case of a non-abbreviation followed by an end-of-sentence punctuation.

[1] Cf. Schmid (2000:1).

The starting point is the *log likelihood ratio* (log λ, Dunning 1993).
If the null hypothesis *(H_0)* – as given in (4) – expresses that the occurrence of a period is independent of the preceding word *w*, the alternative hypothesis *(H_A)* in (5) assumes that the occurrence of a period is not independent of the occurrence of the word preceding it.

(4)H_0: $P(\bullet|w) = p = P(\bullet|\neg w)$
(5)H_A: $P(\bullet|w) = p_1 \neq p_2 = P(\bullet|\neg w)$

The log λ of the two hypotheses is given in (6). Its distribution is asymptotic to a χ^2 distribution and can hence be used as a test statistic (Dunning 1993).

$$(6) \quad \log \lambda = -2 \log \left(\frac{L(H_0)}{L(H_A)} \right)$$

1 Problems for an unscaled log λ approach

Although log λ identifies collocations much better than competing approaches (Dunning 1993) in terms of its *recall*, it suffers from its relatively poor *precision rates.* As is reported in Evert et al. (2000), log λ is very likely to detect all collocations contained in a corpus, but as more collocations are detected with decreasing log λ, the number of wrongly classified items increases. The table in (7) is a sample from the *Wall Street Journal (1987).*[2] According to the asymptotic χ^2 distribution all the pairs given in (7) count as candidates for abbreviations, since the log λ value is higher than 7.88.[3] If a candidate should not be analyzed as an abbreviation, this is indicated in boldface. Some of the 'true' abbreviations are either ranked lower than non-abbreviations or receive the same log λ values as non-abbreviations. This means that an unmodified log λ approach to the detection of abbreviations will produce many errors and thus cannot be employed.

(7) *Candidates for abbreviations from WSJ (1987)*

Candidate	*C(w, •)*	*C(w, ¬•)*	*log λ*
L.F	5	0	29.29
N.H	5	0	29.29
holiday	7	4	27.02
direction	8	8	25.56
ounces	4	0	23.43
Vt	4	0	23.43
debts	7	7	22.36
Frankfurt	5	2	21.13
U.N	3	0	17.57
depositor	3	0	17.57

2 Scaling log likelihood ratios

Since a pure log λ approach falsely classifies many non-abbreviations, we use log λ as a basic ranking, which is scaled by several factors. Measuring their effect in terms of precision and recall on a training corpus from WSJ has experimentally developed these factors.[4] The result of the scaling operation is a much more compact ranking of the true positives in the corpus at the top of the candidate list. The effect of the scaling methods on the data presented in (7) is illustrated in (8).
After applying the scaling factors, the asymptotic relation to the χ^2 distribution cannot be retained. The threshold value for the classification as an abbreviation is hence no longer determined by the χ^2 distribution, but determined on the basis of the classification results derived from the training corpus. The scaling factors, once they have been determined on the basis of the training corpus, have not been modified any further. In this sense, the method described here can be characterized as a corpus-filter method, where the same given corpus is used to filter the initial results obtained from it (cf. Grefenstette 1999:128f.).

[2] As distributed by ACL/DCI. We have removed all annotations from the corpora before processing them so that no sentence boundary information is left

[3] This is the corresponding χ^2 value for a confidence degree of 99.99 %.

[4] The training corpus had a size of 6 MB.

(8) *Result of applying scaling factors*

Candidate	*log λ*	*S(log λ)*
L.F	29.29	216.43
N.H	29.29	216.43
holiday	27.02	0.03
direction	25.56	0.00
ounces	23.43	3.17
Vt	23.43	173.14
debts	22.36	0.00
Frankfurt	21.13	0.01
U.N	17.57	17.57
depositor	17.57	0.04

In the present setting, applying the scaling factors to the training corpus has led to a threshold value of 1.0. This value was determined manually by looking at the ranking and finding the best possible cut-off. We used this value for all experiments described in section 4. Hence, a value above 1.0 allows a classification of a given pair as an abbreviation, while a value below that leads to an exclusion of the candidate. An ordering of the candidates from table (8) is given in (9), where the threshold is indicated through the dashed line.

(9) *Ranking according to S(log λ)*

Candidate	*log λ*	*S(log λ)*
L.F	29.29	216.43
N.H	29.29	216.43
Vt	23.43	173.14
Thurs	29.29	29.29
U.N	17.57	17.57
ounces	23.43	3.17
depositor	17.57	0.04
holiday	27.02	0.03
Frankfurt	21.13	0.01
direction	25.56	0.00
debts	22.36	0.00

As can be witnessed in (9), the scaling methods are not perfect. In particular, *ounces* is still wrongly considered as an initial element of an abbreviation, pointing to a weakness of the approach, which will be discussed in section 7.

3 The scaling factors

We have employed three different scaling factors, as given in (10), (11), and (12).[5] Each scaling factor one after the other is applied to the log λ of a candidate pair. The weighting factors are formulated in such a way as to allow a tension between them (cf. section 3.4). The effect of this tension is that an increase following from one factor may be cancelled out or reduced by a decrease following from the application of another factor, and vice versa.

(10) $S_1(\log \lambda)$: $\log \lambda \cdot e^{C(word, \bullet)/C(word, \neg\bullet)}$.

(11) $S_2(\log \lambda)$: $\log \lambda \cdot \frac{C(word,\bullet) - C(word,\neg\bullet)}{C(word,\bullet) + C(word,\neg\bullet)}$.

(12) $S_3(\log \lambda)$: $\log \lambda \cdot {}^{1}\!/_{e^{\text{length of word}}}$.

3.1 Ratio of occurrence: S_1

By employing scaling factor (10), the log λ is additionally weighted by the ratio between the occurrence of pairs of the form *(word, •)* and the occurrence of pairs of the form *(word, ¬•)*. If events of the second type are either rare or at least lower than events of the first type, the scaling factor leads to an increase of the initial log λ value.[6]

3.2 Relative difference: S_2

The second scaling factor is a variation of relative *difference*. Depending on the figures of *C(word, •)* and *C(word, ¬•)*, its value can be either positive, negative, or 0.

(13) If *C(word, •)* > *C(word, ¬•)*, $0 < S_2 \leq 1$.
(14) If *C(word, •)* = *C(word, ¬•)*, $S_2 = 0$.
(15) If *C(word, •)* < *C(word, ¬•)*, $-1 \leq S_2 < 0$.

If *C(word, ¬•) = 0*, S_2 reaches a maximum of 1. Hence, S_2 in general leads to a reduction of the initial log λ value. S_2 also has a significant effect on log λ if the occurrence of *word* with • equals the occurrence of *word* without •. In this case, S_2 will be 0. Since the log λ values are multiplied with each scaling factor, a value of 0 for S_2 will lead to a value of 0 throughout. Hence the pair *(word, •)* will be excluded from being an

[5] The use of *e* as a base for scaling factors S_1 and S_3 reflects that log λ can also be expressed as H_A being $e^{\log \lambda/2}$ more likely than H_0 (cf. Manning/Schütze 1999:172f.).

[6] If *C(word, ¬•) = 0*, $S_1(\log \lambda) = \log \lambda \cdot e^{C(word,\bullet)}$, reflecting an even higher likelihood that the pair should actually count as an abbreviation.

abbreviation. This move seems extremely plausible: if *word* occurs approximately the same time with and without a following •, it is quite unlikely that the pair *(word, •)* forms an abbreviation.[7] Similarly, the value of S_2 will be negative if the number of occurrences of *word* without • is higher than the number of occurrences of *word* with •. Again, the resulting decrease reflects that the pair *(word, •)* is even more unlikely to be an abbreviation. In fact, such candidates are ruled out with the present cut-off value of 1.0.

Both the relative difference (S_2) and the ratio of occurrence (S_1) allow a scaling that abstracts away from the absolute figure of occurrence, which strongly influences log λ.[8]

3.3 Length of abbreviations: S_3

Scaling factor (12), finally, leads to a reduction of log λ depending on the length of the word that precedes a period. This scaling factor follows the idea that an abbreviation is more likely to be short.

3.4 Interaction of scaling factors

As was already mentioned, the scaling factors can interact with each other. Consequently, an increase by one factor may be reduced by another one. This can be illustrated with the pair (U.N, •) in (13). The application of the scaling factors does not change the value of the initial log λ calculation.

(13) $S_1(\text{U.N}, \bullet) = e^3$,
$S_2(\text{U.N}, \bullet) = 1$,
$S_3(\text{U.N}, \bullet) = \frac{1}{e^3}$

Since the length of *word* actually equals its occurrence together with a •, and since U.N never occurs without a trailing •, S_1 leads to an increase by a factor of e^3, which however is fully compensated by the application of S_3.

4 Experiments

The scaling methods described in section 3 have been applied to test corpora from English (Wall Street Journal, WSJ) and German (Neue Zürcher Zeitung, NZZ). The scaled log λ was calculated for all pairs of the form (word, •). The test corpora were annotated by our tokenizer in the following fashion: If the value was higher than 1, the tag <A> was assigned to the pair. All other candidates were tagged as <S>.[9] The automatically tagged corpora were compared with their hand-tagged references.

(14) Annotation for test corpora

Tag	*Interpretation*
<S>	End-of-Sentence
<A>	Abbreviation
(<A><S>)	(Abbreviation at end of sentence)

We have chosen three different types of test corpora: First, we have used two test corpora of an approximate size of 2 and 6 MB, respectively. The WSJ corpus contained 19,771 candidates of the form (word, •); the NZZ corpus contained 37,986 such pairs. Second, we have tried to determine the sensitivity of the present approach to data sparseness. Hence, the approach was applied to ten individual articles from each WSJ and NZZ. For English, these articles contained between 7 and 26 candidate pairs, for German the articles comprised between 16 and 52 pairs. Third, we converted the two large test corpora to all upper case and all lower case text. We then tested our approach on them in order to see whether it would be affected by the loss of capitalization information.

In all experiments, we used the simple algorithm (15) as a baseline for comparison.

[7] Obviously, this assumption is only valid if the absolute number of occurrence is not too small.

[8] As an illustration, consider the pairs *(outstanding, •)* and *(Adm, •)*. The first pair occurs 260 times in our training corpus, the second one 51 times. While *(outstanding, ¬•)* occurs 246 times, *(Adm, ¬•)* never occurs. Still, the log λ value for *(outstanding, •)* is 804.34, while the log λ value for *(Adm, •)* is just 289.38, reflecting a bias for absolute numbers of occurrence.

[9] A tokenizer should treat pairs that have been annotated with <A> as single tokens, while tokens that have been annotated with <S> should be treated as two separate tokens. Three-dot-ellipses are currently not considered. Also <A><S> tags, i.e. abbreviations at the end of the sentence, are not considered in the experiments (cf. section 7). We did not count them in the evaluation.

(15) 1. tag • as <A> when followed by *lowercase letter, a number,* - ` *;* , or ´
2. tag • as <S> in all other cases

We assume that correctly classified end-of-sentence marks count as true positives in the evaluation of our experiments. In the following tables, we have reported two measures: first, the *error rate,* which is defined in (16), and second, the *F measure* (cf. van Rijsbergen 1979:174), which is a weighted measure of *precision and recall,* as defined in (17).[10]

(16) *Error rate*[11]

$$\frac{C(<A> \rightarrow <S>) + C(<S> \rightarrow <A>)}{C(\textit{all candidates})}$$

(17) *F measure:* $\frac{2PR}{(R+P)}$

4.1 Results of first experiment

The results of the classification process for the larger files are reported in table (18). *F(B)* and *F(S)* are the F measure of the baseline, and the present approach, respectively. *E(B)* is the error rate of the baseline, and *E(S)* is the error rate of the scaled log λ approach.

(18) *Results of classification for large files*

	F(B)	*F(S)*	*E(B)*	*E(S)*
WSJ	88.05	99.05	18.49	1.30
NZZ	95.30	99.11	8.92	1.63

As (18) shows, the application of the scaled log λ leads to significant improvements for both files. In particular, the error rate has dropped from over 18 % to 1.3 % in the WSJ corpus. For both files, the accuracy is beyond 99 %.[12]

[10] Manning/Schütze (1999:269) criticize the use of a*ccuracy* and *error* if the number of *true negatives* – *C(<A> → <A>)* in the present case – is large. Since the number of true negatives is small here, *accuracy* and *error* escape this criticism.

[11] *C(<X> → <Y>)* is the number of X that have been wrongly classified as Y. In(17), P stands for *precision,* and R for *recall.*

[12] It has to be taken into consideration that the method presented here does not make use of any heuristics at all. If some basic heuristics, such as "every token consisting of internal periods and letters is an abbreviation", are included, the error rate drops to approx. 0.5 % for WSJ and NZZ.

4.2 Results of second experiment

The results of the second experiment are reported in table (19) for the articles from the *Wall Street Journal,* and in table (20) for the articles from *the Neue Zürcher Zeitung.* For the articles from WSJ, the scaled log λ approach is somewhat less effective than the baseline approach, although it has a lower error rate for some articles. This is clearly a problem of data sparseness (cf. section 7).

(19) *Results of classification for single articles from WSJ*

	F(B)	*F(S)*	*E(B)*	*E(S)*
WSJ_1	100.00	77.78	0.00	28.57
WSJ_2	89.66	100.00	16.67	0.00
WSJ_3	100.00	100.00	0.00	0.00
WSJ_4	97.3	94.74	3.85	7.69
WSJ_5	80.00	80.00	25.00	25.00
WSJ_6	100.00	92.86	0.00	12.50
WSJ_7	100.00	100.00	0.00	0.00
WSJ_8	100.00	90.00	0.00	14.29
WSJ_9	80.00	72.73	15.38	23.08
WSJ_10	90.91	100.00	14.29	0.00
μ	93.79	90.81	7.52	11.11

(20) *Results of classification for single articles from NZZ*

	F(B)	*F(S)*	*E(B)*	*E(S)*
NZZ_1	95.08	98.31	9.38	3.13
NZZ_2	93.02	97.56	13.04	4.35
NZZ_3	96.97	97.96	5.77	3.85
NZZ_4	96.15	100.00	7.41	0.00
NZZ_5	94.25	98.80	10.64	2.13
NZZ_6	96.84	98.92	6.12	2.04
NZZ_7	97.50	97.37	4.88	4.88
NZZ_8	89.66	100.00	18.75	0.00
NZZ_9	96.97	100.00	5.88	0.00
NZZ_10	93.94	98.41	11.43	2.86
μ	95.04	98.73	9.33	2.32

In general, the articles from NZZ contained fewer abbreviations. Still, the present approach has a much lower average error rate than the baseline approach. One reason is the fact that all nouns are capitalized in German, which is a problem for our baseline approach. Particularly noteworthy are the articles NZZ_1, NZZ_4, and NZZ_8, where the error rate is reduced to 0. In

general, the error rate has been reduced to a fourth.

4.3 Results of third experiment

The results of the third experiment (21) show that our approach is equally well suited to process single case corpora (all upper case or all lower case). The loss of capitalization information leads only to a very small increase in error, slightly greater for the German corpus, which may result from the greater importance of capitalization in German because of its general noun capitalization rule.

(21) *Results for single case corpora*

	F(B)	*F(S)*	*E(B)*	*E(S)*
WSJ (lc)	93.60 %	98.99 %	9.28 %	1.39 %
NZZ (lc)	33.22 %	98.92 %	77.39 %	1.98 %
WSJ (UC)	81.11 %	98.88 %	31.78 %	1.39 %
NZZ (UC)	94.77 %	98.79 %	9.94 %	2.20 %

As can be witnessed in (21), the baseline approach is not suited for single case texts at all producing a highest error rate of over 75 % for the German corpus and over 30 % for the English corpus.

5 Comparison the other approaches

The task of period disambiguation has until recently not attracted much attention. In the past, heuristics and corpus-specific rules in the form of regular grammars were employed for sentence boundary detection. One example is the baseline approach we used for comparison. While it consists of a few simple rules that are easy to develop and to implement, its performance is relatively poor. With such heuristic rule-based approaches, many rules, which are often corpus-specific, are necessary to reach accuracy comparable to log λ or related approaches (cf. e.g. the Alembic System[13] with over 100 handcrafted rules). Grefenstette (1999) proposes several related algorithms. Using a lexicon, a list of abbreviations and regular expressions, he achieves an accuracy of 99.07 % on the Brown corpus. But the major drawback remains: All these approaches have to be developed specifically for one language, one domain, or even one corpus. Another class of algorithms uses machine-learning techniques such as neural networks or decision trees to classify each occurrence of a period. One example is the SATZ system (Palmer & Hearst 1997) which uses part-of-speech (POS) information of the three tokens preceding and the three tokens following a period as input to a neural network or a decision tree. The lowest error rate reported for SATZ is 1 % misrecognized periods on the WSJ corpus, and 0.7 % on a German news corpus. This system is adaptable to different languages and domains, because it can be retrained for different corpora. It is also relatively robust achieving an error rate of 3.3 % on a single case version of the WSJ test corpus. The German single case results did not differ significantly from the results for German mixed case texts. Even OCR scanned texts are processed with relatively high accuracy. The disadvantage of most machine-learning approaches including SATZ is that they depend on resources such as lexica and POS tagged training corpora that are not available for all languages and have to be created manually. Moreover, the SATZ system's performance decreases significantly to an error rate of 4.9 % on the WSJ, when the lexicon it uses does not comprise abbreviations.

All the different approaches above have one thing in common. They rely on the token itself and its context to disambiguate it. They could thus be called "token-based" algorithms. The log λ approach in contrast looks at the distribution of the possible abbreviations in the whole corpus and is thus a "type-based" approach. Two other largely type-based algorithms have been proposed by Mikheev (2000) and Schmid (2000). Mikheev combines the detection of abbreviations, sentence boundaries and proper names in one system. For the detection of abbreviations, he uses a combination of token-based heuristics (e.g. a token consisting of letters and internal periods is an abbreviation) and what he calls the "document-centered approach". For each type that could be an abbreviation, he searches for instances that appear in unambiguous contexts, e.g. before a lower case word. He then uses these unambiguous instances to disambiguate the instances in ambiguous contexts. His system achieves a remarkable error rate of 0.45 % (with abbreviation list) and 1,41 % (without abbreviation list) on the WSJ. It still needs a training

[13] Aberdeen et al. (1995)

corpus and some word lists, but no manual effort is necessary as these can be generated automatically. Because of the importance it attaches to capitalization, it is not suited for single case text and has problems with German noun capitalization. Some of the heuristics it uses are also too language specific in our opinion, e.g. the assumed maximal length of abbreviations of 4 letters in front of the period.

Schmid's approach, like the present one, is only in need of the corpus it is supposed to tokenize. It uses statistical information taken from the local context and the whole corpus to compute the probability of a period being an abbreviation and/or a sentence mark. The calculations are much more complex than our simple scaling factors. The method is language and domain independent. It still relies very much on capitalization information. On the whole WSJ corpus, it achieves an error rate of 0.38 %.

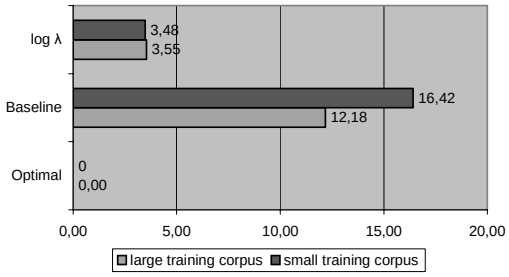
As our system currently only decides whether a token is an abbreviation or not and we have not counted abbreviations at the end of sentences in the evaluation, it is not directly comparable to the other approaches. Our current results, however, are very promising also for integration into a sentence boundary detection system. Without using any heuristics, the log λ approach achieves very low error rates. It is language and corpus independent and does not require any manual work or extra resources. Moreover, it does not depend on capitalization information. It is thus robust, simple, fast, and, efficient.

6 The impact of tokenization on tagging

In order to determine the relevance that tokenization has for subsequent processing steps like tagging, we trained the statistical TnT-Tagger (Brants 2000) on two correctly tokenized and tagged training corpora (5.8 million and 21.6 million tokens from the WSJ respectively). These corpora were assembled from those individual articles containing the highest number of abbreviations. We then prepared three different versions of the test corpus (2 million tokens from WSJ). In the first version, we kept the original classification of the periods (end-of-sentence marks are tagged as • in the WSJ corpus). The second version was tokenized with the system outlined in this paper, and the third version was tokenized with the following (different) baseline approach: All periods excluding internal periods, i.e. decimal points and periods inside of abbreviations, were regarded as sentence marks and separated from the preceding token. We then tagged the three different test corpora with the TnT-tagger, which had been trained on the two training corpora.

The results show that tokenization has a considerable influence on the number of tagging errors. The increase in the percentage of tagging errors is illustrated in (22).

(22) *Tokenization and increase of tagging error*



The log λ approach increased the tagging error rate by approx. 3.5 % compared to the optimal tokenization. The baseline approach resulted in an average increase of approx. 14.3 %.

Our approach reduces the tagging errors resulting from incorrect tokenization by approx. 72 % compared to the baseline.

7 Weaknesses and future steps

We have noted in section 2 that the scaling factors do not lead to a perfect classification. This is particularly reflected in the application of S(log λ) to WSJ_1 and NZZ_7, which actually show the same problem: In the training corpus from WSJ, *ounces* was always followed by •. In WSJ_1, the word *said* was always followed by •, and this also happened in NZZ_7 for *kann*. Without the inclusion of additional metrics, non-abbreviations that exclusively occur at the end of sentences are wrongly classified. As the probability of a non-abbreviation occurring solely in front of a period generally decreases with its total number of occurrence, one possibility to avoid this problem would be to use other metrics as a back-off when either corpus size or number of occurrences are too low. The table in (23) illustrates, however, that the error rate for false

negatives drops significantly if plausible corpus sizes are considered.[14]

(23) *False negatives (f.n.) and corpus size*

	<S>	f.n. = <S> → <A>	Error %
NZZ	34,400	61	1.63
WSJ	13,488	36	1.3
NZZ_7	41	2	4.88
WSJ_1	14	4	28.57

We have also ignored abbreviations occurring at the end of the sentence. The question whether an abbreviation is also the end of the sentence has to be decided for each individual token. We are currently developing statistical algorithms that can handle this problem without having to be trained on a manually annotated corpus.[15]
Another problem of our type-based approach are abbreviations that are homographic with non-abbreviations, e.g. '*no.*', which can either represent the abbreviation for *number* or the negative particle '*no*'. Our algorithm is not able to distinguish such homographs. Fortunately, such cases are rather rare.

Additional methods for the detection of abbreviations could be integrated into our approach. As abbreviations often consist of letter combinations which are highly unlikely in normal words, one possibility could be to use 'phonotactic" information in the form of letter n-grams to calculate the probability of a token being an abbreviation.
Furthermore, we believe that periods following numbers and initials, i.e. single capital letters, should be treated separately with different methods in order to decrease the error rate even further.[16]

Conclusion

We have presented an accurate and comparatively simple method for the detection of abbreviations, which makes use of scaled log likelihood ratios. Experiments have shown that the method works very well with large files and showed a performance comparable to simple heuristic algorithms for very small corpora with sparse data. The method does not depend on manually annotated training corpora and is very fast even for large corpora. We expect further improvements once additional classification schemata have been integrated.

[14] If, however, in some language, certain function words always have to occur at the end of the sentence, this could be a problem for our type-based approach.

[15] cf. also Schmid (2000)

[16] cf. also Schmid (2000)

References

Aberdeen, J., J. Burger, D. Day, L. Hirschmann, P. Robinson, and M. Vilain (1995) *Description of the alembic system used for muc-6.* In: The Proceedings of the Sixth Message Understanding Conference (MUC-6). Morgan Kaufmann.

Brants, Th. (2000) *TnT - A Statistical Part-of-Speech Tagger*. In: Proceedings of the Sixth Applied Natural Language Processing Conference ANLP-2000, Seattle, WA

Dunning, T. (1993) *Accurate methods for the statistics of surprise and coincidence.* Computational Linguistics, 19/1, pp. 61—74.

Evert, S., U. Heid and W. Lezius (2000) *Methoden zum qualitativen Vergleich von Signifikanzmaßen zur Kollokationsidentifikation.* ITG Fachbericht 161, pp. 215—220.

Grefenstette, G. (1999) *Tokenization.* "Syntactic Wordclass Tagging", H. van Halteren, ed., Kluwer Academic Publishers, pp. 117—133.

Liberman, M.Y. and K.W. Church (1992) *Text analysis and word pronunciation in text-to-speech synthesis.* In "Advances in Speech Signal Processing", S. Furui & M.M. Sondhi, ed., M. Dekker Inc., pp. 791—831.

Manning, C.D. and H. Schütze (1999) *Foundations of statistical natural language processing.* The MIT Press, Cambridge/London.

Mikheev, A. (2000) *Tagging Sentence Boundaries*. In: NACL'2000 (Seattle) ACL, pp. 264-271

Palmer, D.D. and M.A. Hearst (1997) *Adaptive multilingual sentence boundary disambiguation.* Computational Linguistics, 23/3, pp. 241—267.

Schmid, H. (2000) *Unsupervised Learning of Period Disambiguation for Tokenisation.* Internal Report, IMS, University of Stuttgart.

van Rijsbergen, C.J. (1979) *Information Retrieval*. Butterworths, London.