Word and sentence segmentation in German: Overcoming idiosyncrasies in the use of punctuation in private communication

Kyoko Sugisaki
German department, University of Zurich
Schönberggasse 9, 8001 Zurich, Switzerland
sugisaki@ds.uzh.ch

Abstract

In this paper, we present a segmentation system for German texts. We apply conditional random fields (CRF), a statistical sequential model, to a type of text used in private communication. We show that by segmenting individual punctuation, and by taking into account freestanding lines and that using unsupervised word representation (i.e., Brown clustering, Word2Vec and Fasttext) achieved a label accuracy of 96% in a corpus of postcards used in private communication.

1 Introduction

In tokenisation and sentence segmentation, a text is segmented into tokens and sentences. Word and sentence segmentation are the core components of NLP pipelines. Based on text segmentation, part of speech (POS) tagging and parsing, among other tasks, are performed.

In German texts, segmentation is de facto to classify sentence punctuation, such as periods, question marks and exclamation marks, into two categories: (A) the ends of sentences, and (B) others, such as components of abbreviations (e.g. *evtl.*, *eventuell* 'possibly'), proper names (e.g. Sat.1), numbers (e.g., 13.000) and so on. In the case of (A), the punctuation is separated from space-delimited tokens and analysed as individual tokens. In the case of (B), the punctuation constitutes a token with the preceding characters. Therefore, space-delimited tokens are not segmented further. In rare cases, punctuation that is used to mark the end of a sentence (i.e., category [A]) is a part of the token (i.e. category [B]) at the end of a sentence.[1]

Traditionally, German text segmentation systems are based on rules that contain a list of abbreviations.[2] A rule-based approach to the segmentation of German texts (Remus et al., 2016; Proisl and Uhrig, 2016) is reasonable considering the complexity of the task. In a newspaper corpus (Tübinger Baumbank des Deutschen/Zeitungskorpus (Tüba-D/Z) v. 10, henceforth TüBa10, there are 1.787.801 tokens and 95.595 sentences, described in Telljohann et. al (2012)), about 91% of sentence boundaries are punctuation such as periods, colons, semicolons and commas (Table 1). The remaining sentences end with a word. As expected, periods are the most frequently used at the ends of the sentences in TüBa10 (about 77%, cf. Table 1). Most of the periods (about 84% of all periods) are used to mark the end of a sentence (Table 1). The remaining periods are parts of tokens (i.e., category [B]), of which 68 types are identified in the corpus. If we exclude token types that we can simply handle with regular expressions – that is, those with an alphabet, number, email address, web link and ellipsis – there are 27 types of abbreviations and proper names. These exceptions can be handled reasonably by listing the abbreviations and proper names.[3]

However, the task of text segmentation is not trivial if we address the following dependencies (Palmer, 2000): 1) language dependence, 2) corpus dependence and 3) application dependence. Thus, the segmentation of multi-lingual texts (Jurish and Würzner, 2013; Kiss and Strunk, 2006) is not rule-based but statistical. Corpus dependence involves a wide range of text types that have various linguistic features. Lastly, the definitions of words and sentences depend on the NLP application: for example, in a machine translation, a German compound is better split into individual morphemes (El-Kahlout

[1]For instance, 176 (0.18%) in 95.595 sentences belong to the third category in TüBa10. An example is *usw.* at the end of a sentence.

[2]Helmut Schmid's tokenizer in TreeTagger: `http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/`; Stefanie Dipper's system: `https://www.linguistics.ruhr-uni-bochum.de/~dipper/resources/tokenizer.html`

[3]However, lists of abbreviations are never complete, and need to be extended, when we use out-of-domain data.

Last tokens of sentences	ranking:tokens (frequency, %)
1:Period (73904, 77.31%), 2:double quotation (3849, 4.03%), 3:question mark (2921, 3.06%), 4:colon (2369, 2.48%), 5:exclamation mark (682, 0.71%), 6:semicolon (634, 0.66%), 7:parentheses(393, 0.41%), 8:ellipsis (329, 0.34%), 11:guillemet (59, 0.06%), 21:comma (26, 0.03%), 22:square bracket(26, 0.03%), 25:hypen (24, 0.03%), 37:single quote (18, 0.02%), 212:slash (6, 0.01%), else (10355, 10.83%)	
Ambiguity of punctuations	tokens(A: frequency of case (A), the total number of the character, AMB(iguity):A/total(%)
Period (A:73904 + 329*3, total:88938, AMB:84.20%), double quotation (A:3849, total:42468, AMB:9.06%), question mark (A:2921, total:3536, AMB:82.60%), colon (A:2369, total:11522, AMB:20.56%), exclamation mark (A:682, total:1424, AMB:47.89%), semicolon (A:634, total:, AMB:%), parentheses (A:393, total:5999, AMB:6.55%), guillemets (A:59, total:369, AMB:15.98%), comma (A:26, total:102425, AMB:0.02%), square bracket (A:26, total:75, AMB:34.66%), hypen (A:24, total:29863, AMB:0.08%), single quote (A:18, total:730, AMB:2.46%), slash (A:6, total:2065, AMB:0.29%)	

Table 1: Use of punctuations (TüBa10)

and Yvon, 2010).

In this work, we focus on the development of a German text segmentation system that deals with the issue of corpus dependence in Palmer's term. More specifically, it has been observed - e.g. by Giesbrecht and Evert (2009) for part-of-speech-tagging and by Gildea (2001) for parsing - that a statistical model usually works for the domain and text types it has been trained for, but leaves to desire when applied to other domains and text types. In this work, we undertake domain adaptation in text segmentation, in particular, with a target domain - texts written in private communication. Typically, these texts contain many deviations from standard orthography, including idiosyncrasies in capitalisation and punctuation.

In this paper, we train text segmentation models (conditional random fields) on TüBa10 (Section 4) and test them on an example of a text used in private communication: a postcard corpus[4] (*Ansichtskartenkorpus*, 'picture postcard corpus', henceforth ANKO) (Section 5). Sections 2 and 3 provide the analysis of the use of punctuation in private communication, and describe our text segmentation system.

2 Use of Punctuation

In German, punctuation segments a text into sentences, and a sentence is segmented into words by spaces. However, these rules of thumb are not applicable in the following cases of sentence segmentation: (1) punctuation that is a part of a token with preceding characters (e.g., abbreviations); and (2) punctuation is absent. Case (1) was discussed in Section 1. Case (2) occurs because of freestanding lines. Freestanding lines typically end with a line break with a wide blank space or extra line spacing, and often do not end with a punctuation. Examples are titles, subtitles, addresses, dates, greetings, salutations and signatures (Official German Orthography, 2006). In private communication, the rules for freestanding lines are also applied to the end of paragraphs. In addition, the following usage is common to the punctuation in a private communication: (a) repeated punctuation (e.g. , *!!!, ???,* ......) in order to emphasise words, phrases and sentences; and (b) the use of emotional pictograms that are typically composed of punctuation (e.g. :), ;-)) (cf. Bartz et. al (2013)).

3 Conditional Random Fields (CRF)-Based Text Segmentation

We develop a CRF-based German text segmentation system that can be applied to the types of texts used in private communication (cf. Section 2). We focus on tokenisation with punctuation and sentence segmentation. In this section, we briefly introduce CRF and define the notion of a sequence and a set of features used in the task.

3.1 Conditional Random Fields

CRF (Lafferty et al., 2001; Sutton and McCallum, 2011) is a random field (also known as undirected graph or Markov network) for conditional probability $P(y_{1:n}|x_{1:n})$, where $x_{1:n}$ is an input sequence $x_1 \ldots x_n$ and $y_{1:n}$ is an output sequence $y_1 \ldots y_n$. To calculate the conditional probability, CRF makes use of the maximum entropy model and normalizes the probability globally in a sequence:

$$P(y_{1:n}|x_{1:n}) = \frac{1}{Z(x_{1:n})} \exp\left(\sum_{n=1}^{N}\sum_{d=1}^{D} w_d f_d(x_{1:n}, y_n, y_{n-1}, n)\right)$$

$$Z(x_{1:n}) = \sum_{y_{1:n}} \exp\left(\sum_{n=1}^{N}\sum_{d=1}^{D} w_d f_d(x_{1:n}, y_n, y_{n-1}, n)\right)$$

3.2 Sequence

The CRF model learns the parameters and decodes the output based on a given sequence of input units. In our classification task, a text is a sequence of input units. We use the term *unit* to denote each atomic element in the CRF in order to differentiate

[4]The corpus will be released in `https://linguistik.zih.tu-dresden.de/ansichtskarten`.

it from the term *token* or *word*. We create units by splitting texts using white spaces and by separating punctuation from the attached characters. We then classify the input units into three categories: the beginning (B), intermediate (I) and end (E) of sentences. Using this notation, the chunk of a token is also marked.

We investigate how flexibly punctuation should be handled in order to be robust for domain difference, and the importance of punctuation in text segmentation. To this end, we create three types of sequences by deliberately handling the punctuation listed in Table 1 in the following three ways:

(a): Punctuation before a white space is regarded as a unit. For example, *z.B.* (abbreviation of *zum Beispiel* 'for example') consists of two units: *z.B* and .

(b): Punctuation is regarded as a unit regardless of a white space. Accordingly, *z.B.* consists of four units: *z*, ., *B* and .

(c): All punctuation is removed if it is followed by a white space. Accordingly, *z.B.* consists of one unit: *z.B*

Variant (a) is a setting in which the white space is well placed, and it follows standard German orthographic rules. In Variant (b), every punctuation mark is individually handled, which is expected to provide flexibility in orthographical deviations. In Variant (c), punctuation is missing in the input text.

3.3 Features

Features are key linguistic indicators that may be useful in the segmentation of sentences. In this work, we use three types of features to handle orthographic variations and unknown words in variations of types of text: forms, POS and unsupervised word representations. The following subsections describe each feature in detail (Table 2 in Appendix).

Form Forms of units are integrated as three features: (1) unit, (2) character types of unit and (3) normalised unit. For the first feature, units are extracted as they are. In the second feature, units are categorised into alphabetic, numeric, types of special characters, and their combinations. For the third feature, units are changed to lower case.

Part of speech POS is used as a feature in two ways: fine-grained original Stuttgart-Tübingen-TagSet (STTS, Schiller et. al (1999)) or coarse POS (CPOS), which are shown in Table 2. These two features are extracted automatically using the TreeTagger (Schmid, 1999).

Brown clustering For the first feature of unsupervised word clustering, the hierarchical classes of Brown clustering (Brown et al., 1992) are exploited. Brown clustering is a bigram-based word clustering that has been successfully integrated to improve parsing (Koo et al., 2008), domain adaptation (Candito et al., 2011) and named entity recognition (Miller et al., 2004). We ran Brown clustering on the normalised tokens (i.e. all lower case) of TüBa10 to build 100 clusters.[5] For the features, we used the first four digits and all digits in the clustering hierarchy. In the data, the grouping of named entity such as person and organization and the part of speech such as noun and verb were captured the most clearly in the first four digits of the clustering.

Word2Vec For the second feature of unsupervised methods, we used k-means clustering in Word2Vec. Word2vec (Mikolov et al., 2013) is another kind of word representation. We ran the Word2Vec on the normalised tokens of TüBa10 to build the models.[6] To operationalise the word-embedding vectors, we further grouped them into 50 K-means clusters.[7] The resulting clusters contained a great deal of named entities.

Fasttext For the third feature of unsupervised methods, we used k-means clustering in Fasttext. Fasttext (Bojanowski et al., 2016) is yet another kind of word representation that takes into account character n-grams (morpheme). We ran Fasttext on the normalised tokens of TüBa10 to build the models.[8] We further grouped them into 200 K-means clusters that contained a large number of German compounds.

4 Experiments

In this experiment, our goal was to develop a text segmentation model that could robustly be applied to domain difference. For the experiment, we used the TüBa10 in form of (a), (b) and (c) (cf. Section 3.2) with various feature configurations, and we trained and tested the CFG models using five-fold cross-validation. In the next section, we evaluate

[5]We used the Brown clustering implemented by P. Liang

[6]For word2vec, we used gensim with parameters CBOW, 200 dimensions, context window 5

[7]For K-means clustering, we used the scikit-learn

[8]We used the fasttext with parameters, CBOW, 200 dimensions, 5 context window, 5 word ngrams

the models by applying them to a postcard corpus to test their robustness for texts generated in private communications.

Single features In the experiment, we used data in the forms of (a), (b) and (c) with single features. First, we trained CRF models in context window 0. The results are shown in columns #1 to #12 of Table 3 in Appendix. Among the features, the character type of unit (#3) - information about capitalisation, and type of punctuation and characters - showed the best performances in sequence types (a) and (b), whereas gold STTS POS tag (#4) showed the best performance in (c). In the unsupervised methods, Brown clustering (#8/9) outperformed Word2Vec (#10) and Fasttext (#11). As expected, the sequence types (a) and (b) achieved higher accuracy than sequence (c) did. For the sequence type (c), that is, the input sequence without punctuation, all individual features did not predict the classes (B) and (E). Thus, punctuation was proven relevant in text segmentation. We extended the set of features in context window 3. However, the accuracy remained the same as in window 0.

Feature combinations To obtain linguistic information effectively in wide contexts, we combined the features in the following two ways: 1) the same features in context 0 and 1; 2) the combination of two features (1a/1b/1c/2bT/3a4/3b/3c) in context 0. In the first setting, which combined all with all features of previous, current and next tokens, the CRF models improved with regard to class (B) and (E) (#13 in Table 3). In the second feature combination (#14), the overall accuracy was similar to that in the set of all single features (#12).

For the evaluation, we trained all single features in context 0 (#12), the combinations of the same features in context window 1 (#13) and those of various features in context window 0 (#14) without using gold POS and CPOS tags. The CRF models achieved accuracies of 0.99, 0.99 and 0.97 on the TüBa in the sequence types (a), (b) and (c), respectively. In the evaluation, we used the feature set for each input sequence type.

5 Evaluation and Conclusion

For the evaluation, we used a test set derived from a corpus of postcards (ANKO). The corpus comprised over 11,000 holiday postcards sent by post to Swiss households from 1950 to the present day. In this work, we used a sub-corpus (545 cards, 3534 sentences and 25096 tokens) that contained cards mainly written in standard German. We manually created three types of input sequences: (I) one with text boundaries; (II) one with text and paragraph boundaries; and (III) one with text, paragraph and discourse boundaries (date, salutation, greeting and signature). We tested the final models as described in the previous section. The results are shown below:

	(I)		(II)		(III)	
	acc	F1(B,I,E)	acc	F1(B,I,E)	acc	F1(B,I,E)
(a)	.89	.73, .94,.72	.91	.80, .95, .77	.94	.88, .97, **.84**
(b)	.91	.76, .95,.44	.93	.82, .97, .61	**.96**	**.90, .98**, .82
(c)	.81	.30, .90,.42	.83	.40, .90, .55	.86	.50, .92, .73

Overall, the sequence type (b) achieved better accuracy than sequence type (a) did, which showed that orthographic deviations could be handled more effectively by segmenting punctuation individually. Clearly, the patterns of punctuation were more generally captured in (b). Furthermore, the input text type (III) achieved high accuracy. These results indicate that the annotation of a corpus with paragraphs and freestanding lines is relevant in improving the quality of the segmentation of texts used in private communication. Still, it was difficult to predict text segments without having punctuations (c) on a type of text different from the training data.[9]

As comparison, we tested a sentence segmentation system PUNKT (Kiss and Strunk, 2006).[10] PUNKT is based on unsupervised methods and designed for multi-lingual text segmentation. We tested PUNKT on our ANKO test set type (III). PUNKT achieved a F1 score of 0.79 with precision 0.71 and recall 0.9. In contrast, our sentence segmentation system achieved a F1 score of 0.95 with precision 0.94 and recall 0.96, using the input format (b) and (III), that is, the best input format for tokenization.

In conclusion, we presented our German text segmentation system for texts in private communication. In future work, we will extend our text segmentation system on historical German texts.

Acknowledgments

This work has been funded under SNSF grant 160238. We thank all the project members, Heiko Hausendorf, Joachim Scharloth, Noah Bubenhofer, Nicolas Wiedmer, Selena Calleri, Maaike Kellenberger, David Koch, Marcel Naef, Josephine Obert, Jan Langenhorst, Michaela Schnick to support our

[9]Our text segmentation system (GETS) is available: `https://sugisaki.ch/tools`

[10]We used the NLTK module PUNKT.

work. We thank two anonymous reviewers for their valuable suggestions.

References

Thomas Bartz, Michael Beißwenger, and Angelika Storrer. 2013. Optimierung des Stuttgart-Tübingen-Tagset für die linguistische Annotation von Korpora zur internetbasierten Kommunikation: Phänomene, Herausforderungen, Erweiterungsvorschläge. *JLCL*, 28:157–198.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Peter F Brown, Peter V Desouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Marie Candito, Enrique Henestroza Anguiano, and Djamé Seddah. 2011. A word clustering approach to domain adaptation: Effective parsing of biomedical texts. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 37–42.

Ilknur Durgar El-Kahlout and François Yvon. 2010. The pay-offs of preprocessing for German-English statistical machine translation. In *Proceedings of the 7th International Workshop on Spoken Language Translation (IWSLT)*, pages 251–258.

Eugenie Giesbrecht and Stefan Evert. 2009. Is part-of-speech tagging a solved task? An evaluation of POS taggers for the German web as corpus. In *Proceedings of the Fifth Web as Corpus Workshop (WAC5)*, pages 27–35.

Daniel Gildea. 2001. Corpus variation and parser performance. In *Proceedings of the EMNLP*, pages 167–202.

Bryan Jurish and Kay-Michael Würzner. 2013. Word and sentence tokenization with Hidden Markov Models. *JLCL*, 28:61–83.

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

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of the ACL/HLT*, pages 595–603.

John Lafferty, Andrew McCallum, and Fernando C.N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning 2001*, pages 282–289.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proceedings of HLT-NAACL*, volume 4, pages 337–342.

David D. Palmer, 2000. *Tokenisation and Sentence Segmentation*. CRC Press.

Thomas Proisl and Peter Uhrig. 2016. Somajo: State-of-the-art tokenization for german web and social media texts. In *Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*, pages 57—62.

Steffen Remus, Gerold Hintz, Darina Benikova, Thomas Arnold, Judith Eckle-Kohler, Christian M. Meyer, Margot Mieskes, and Chris Biemann. 2016. EmpiriST: AIPHES Robust Tokenization and POS-Tagging for Different Genres. In *Proceedings of the 10th Web as Corpus Workshop*, pages 106–114.

Anne Schiller, Simone Teufel, Christine Stöckert, and Christine Thielen. 1999. Guidelines für das Tagging deutscher Textcorpora mit STTS (Kleines und großes Tagset). Technical report, Universität Stuttgart, Universität Tübingen, Stuttgart, Germany.

Helmut Schmid. 1999. Improvements in part-of-speech tagging with an application to German. In *Natural Language Processing Using Very Large Corpora*, volume 11 of *Text, Speech and Language Processing*, pages 13–26. Kluwer Academic Publishers, Dordrecht.

Charles Sutton and Andrew McCallum. 2011. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373.

Heike Telljohann, Erhard W. Hinrichs, Kübler Sandra, Zinsmeister Heike, and Beck Kathrin. 2012. Stylebook for the Tübingen treebank of written German (TüBa-D/Z). Technical report, Universität Tübingen.

6 Appendix

1a	**Unit**
1b	**Character type of unit**: unit form is categorised into the following classes: All characters are alphabetic, and (A) consist of just one alphabet or (B) are absent of vocal (e.g. *lg,hrzl*) or (C) all letters are uppercase or (D) only first letter is uppercase or (E) else. Or all characters are (F) numbers or (G) alphanumeric. For punctuations, (H) period, (I) comma, (J) question and exclamation mark, (K) colon and semicolon, (L) opening and (M) closing bracket, (N) opening and (O) closing quotation. For mix classes: (P) alphabets and punctuations/other special characters (e.g. *u.s.w, v.a*), (Q) numbers and punctuations/other special characters (e.g. *8.000*), or (R) else.
1c	**Normalized unit**: all lower case
2a	**Fain-grained POS**: STTS tag set; In experiment, 2aG is gold standard, 2aT is TreeTagger output
2b	**Coarse POS**: Nouns, verbs, modifiers of nouns, modifiers of verbs, relative pronouns, other pronouns, articles, prepositions, postpositions, cardinal number, wh words, subordinating/infinitive conjunctions, coordinating conjunctions, spoken language markers, comma and semicolon, colon, period and question and exclamation mark, quotations, brackets, else; In experiment, 2bG is gold standard, 2bT is based on TreeTagger output
3abc	**Unsupervised methods**: Brown clustering, word2vec and fasttext, respectively. In the experiment, brown clustering is used in 4 digits (3a4) and all digits (3aA)

Table 2: Features

#	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	1a	1b	1c	2aG	2aT	2bG	2bT	3a4	3aA	3b	3c	all	all	all
	single features in context window 0												combinations	
(a)	acc:.97* F1(B):.86 F1(I):.99* F1(E):.86	acc:.97* F1(B):.82 F1(I):.98 F1(E):.84	acc:.97* F1(B):.88* F1(I):.99* F1(E):.88*	acc:.96 F1(B):.81 F1(I):.98 F1(E):.83	acc:.97* F1(B):.83 F1(I):.98 F1(E):.85	acc:.96 F1(B):.81 F1(I):.98 F1(E):.82	acc:.97* F1(B):.83 F1(I):.98 F1(E):.85	acc:.97* F1(B):.84 F1(I):.98 F1(E):.85	acc:.97* F1(B):.81 F1(I):.98 F1(E):.83	acc:.91 F1(B):.47 F1(I):.95 F1(E):.49	acc:.89 F1(B):.08 F1(I):.94 F1(E):.11	acc:.98 F1(B):.91 F1(I):.99 F1(E):.92	**acc:.99** **F1(B):.96** **F1(I):1.00** **F1(E):.96**	acc:.98 F1(B):.91 F1(I):.99 F1(E):.91
(b)	acc:.97* F1(B):.82 F1(I):.98* F1(E):.83	acc:.96 F1(B):.79 F1(I):.98* F1(E):.81	acc:.97* F1(B):.85* F1(I):.98* F1(E):.85*	acc:.96 F1(B):.79 F1(I):.98* F1(E):.81	acc:.97* F1(B):.81 F1(I):.98* F1(E):.83	acc:.96 F1(B):.79 F1(I):.98* F1(E):.81	acc:.97* F1(B):.81 F1(I):.98* F1(E):.84	acc:.97* F1(B):.82 F1(I):.98* F1(E):.84	acc:.96 F1(B):.79 F1(I):.98* F1(E):.81	acc:.91 F1(B):.45 F1(I):.95 F1(E):.48	acc:.89 F1(B):.08 F1(I):.94 F1(E):.10	acc:.98 F1(B):.89 **F1(I):.99** F1(E):.90	**acc:.99** **F1(B):.95** **F1(I):.99** **F1(E):.96**	acc:.98 F1(B):.88 **F1(I):.99** F1(E):.90
(c)	acc:.88 F1(B):.00 F1(I):.93 F1(E):.00	acc:.88 F1(B):.00 F1(I):.93 F1(E):.00	acc:.88 F1(B):.00 F1(I):.94* F1(E):.00	acc:.89* F1(B):.21* F1(I):.94* F1(E):.22*	acc:.88 F1(B):.00 F1(I):.93 F1(E):.00	acc:.88 F1(B):.02 F1(I):.93 F1(E):.03	acc:.88 F1(B):.00 F1(I):.93 F1(E):.00	acc:.88 F1(B):.00 F1(I):.93 F1(E):.00	acc:.88 F1(B):.00 F1(I):.93 F1(E):.02	acc:.88 F1(B):.00 F1(I):.93 F1(E):.00	acc:.88 F1(B):.00 F1(I):.93 F1(E):.00	acc:.96 F1(B):.82 **F1(I):.98** F1(E):.85	**acc:.97** **F1(B):.86** **F1(I):.98** **F1(E):.89**	acc:.96 F1(B):.81 **F1(I):.98** F1(E):.83

Table 3: Feature experiments (5-fold cross validation on TüBa10): abbreviations of features listed in TB 2; sequence type (a)(b)(c) described in Section 3.2; acc(uracy) = correctly predicted tokens/the total number of tokens; F1 = 2 * precision * recall/(precision + recall)