Statement Segmentation for German Easy Language Using BERT and Dependency Parsing

Andreas Säuberli[*]
Department of Computational Linguistics
University of Zurich
andreas.saeuberli@uzh.ch

Niclas Bodenmann[*]
Independent researcher
niclas.bodenmann@gmx.ch

Abstract

Texts in Easy Language should contain a low number of statements per sentence, to make information more accessible and comprehensible. The shared task *Statement Segmentation in German Easy Language (StaGE)* aims to automatically identify the number and location of statements in German Easy Language sentences. We present our submission to this task, which combines sequence labeling with dependency parsing. Our approach uses a fine-tuned BERT model to predict the head token of each statement span and expands the span using dependency relations. Our model achieves a mean absolute error of 0.40 in the predicted number of statements and Jaccard index of 0.38 in the statement spans. We discuss the challenges and limitations of the task and outline future research directions.

1 Introduction

Easy Language is a simplified variety of language with the goal of improving information accessibility, e.g., for persons with cognitive disabilities, prelingual hearing impairments, dementia, or aphasia (Maaß, 2020). The draft for DIN SPEC 33429 (DIN, 2023) represents a recent attempt to provide a standardized set of recommendations and guidelines for creating content in German Easy Language. One of these recommendations is that sentences in Easy Language should contain a small number of statements (DIN, 2023, p. 14). While the document does not elaborate on a specific definition of the term *statement*, this has prompted the conception of the shared task on *Statement Segmentation in German Easy Language (StaGE)*[1] (Schomacker et al., 2024). The aim of the shared task is to automatically identify and segment statements in German Easy Language sentences.

This paper describes our submission to the shared task under the team name *StaGE FriGHt*.

2 Tasks, data, and evaluation

The *StaGE* shared task comprises two subtasks:

Subtask 1: Predict the number of statements in a given sentence.

Subtask 2: If there is more than one statement in a sentence, identify the corresponding token spans for each statement.

The shared task defines *statements* in the theoretical framework of valency grammar, which posits that verbs carry obligatory slots that need to be filled in order to form sound sentences. If only the obligatory slots are filled, the sentence only contains one statement. For example, the sentence *She gave him a gift* contains one statement, because only the three obligatory slots of the verb *gave* are filled: the subject *she*, the direct object *a gift*, and the indirect object *him*.

Each additional, optional slot amounts to another statement. Moreover, optional noun modifiers such as adjectives are also considered to be separate statements. For example, the sentence *She gave him a beautiful gift for his birthday* contains three statements: the verb with its obligatory slots (*She gave him a gift*), the optional slot *for his birthday*, and the adjective *beautiful*. In Subtask 2, the goal is to identify the set of tokens that form each statement. Some function words such as articles or conjunctions are not considered part of the statement span.

The training data consists of 2944 manually annotated sentences. A development set with 416 sentences is provided. The test set contains 878

[*]Both authors contributed equally. The author order was randomized: https://www.aeaweb.org/journals/policies/random-author-order/search?RandomAuthorsSearch%5Bsearch%5D=rHNaKZoddapX

[1]https://german-easy-to-read.github.io/statements/

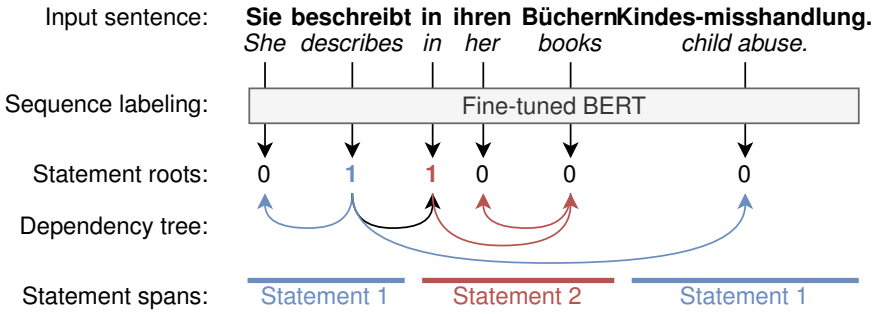

Figure 1: Our approach combines sequence labeling with dependency parsing: First, we use a fine-tuned BERT model to tag the head token of each statement span. We then apply dependency parsing and expand the spans to include all tokens that are dependent on the head token, while avoiding span overlap.

sentences. The data is available on GitHub.[2]

Subtask 1 is evaluated according to the mean absolute error (MAE) in the predicted number of statements. The spans extracted in Subtask 2 are evaluated using character-based F-score (chrF) and Jaccard index.

3 Sequence labeling for discontinuous span segmentation

From a technical perspective, one of the main challenges is that many of the provided statement spans are discontinuous. This means that well-established tagging formats such as the BIO format (Ramshaw and Marcus, 1995), which use a separate label for the beginning of each span, cannot be applied. Previously, several works have extended the BIO format with additional tags to accommodate discontinuous spans (Muis and Lu, 2016; Metke-Jimenez and Karimi, 2016; Tang et al., 2018) or resorted to multilabel classification (McDonald et al., 2005; Tang et al., 2018), adding considerable complexity to the task.

In some cases, spans in the dataset are also overlapping, but it is unclear what guidelines are applied. Compare the two following examples from the training set, both of which use similar syntactic constructions, but only one uses overlapping spans:

(1) *Durch Altenburg fließt der Fluss Pleisse.*
'The river Pleisse flows through Altenburg.'
Statements: [*Durch Altenburg* 'through Altenburg'], [*fließt Fluss* 'flows river'], [*fließt Pleisse.* 'flows Pleisse']

(2) *Sie ist Mitglied in der Partei Die Grünen*
'She is a member of the party The Greens.'
Statements: [*in Partei* 'in party'], [*Die Grünen* 'The Greens']

For simplicity, we restrict ourselves to predicting non-overlapping spans.

Since the statement spans often correspond to syntactical units such as clauses or prepositional phrases, we use syntactic dependency relations to construct statement spans. This allows us to frame the problem as a binary sequence labeling task, simplifying the tagging format compared to previous approaches. It also means that we can handle the two tasks of finding the statements and segmenting the sentence into spans separately.

Specifically, our approach involves the following steps (visualized in Figure 1):

1. We use a fine-tuned BERT model (Devlin et al., 2019) for sequence labeling, classifying for each token in the input sentence whether it is the head of a statement span (i.e., the highest-level token in the dependency hierarchy within that span).

2. We apply dependency parsing using *spaCy* (Honnibal et al., 2020) with the model `de_dep_news_trf`[3] to the input sentence and align the result to the tokenization in the dataset.

3. For each head token we found in step 1 (starting with the lowest-level one in the dependency tree), we expand the statement span around it by following the dependency relations, adding each descendant token to the span. We stop as soon as we reach a token that already belongs to a different span, to avoid overlap. We exclude articles, punctuation, and coordinating conjunctions.

[2] https://github.com/german-easy-to-read/statements/tree/master/data

[3] https://spacy.io/models/de#de_dep_news_trf

Description	Sentence	True spans	Predicted spans
Correct	*Im Jahr 2002 macht Lula wieder mit bei der Prösidenten-wahl.* (sic)	[*Im Jahr 2002*] [*macht Lula wieder mit bei Prösidenten-wahl.*]	[*Im Jahr 2002*] [*macht Lula wieder mit bei Prösidenten-wahl.*]
(Translation:)	*In the year 2002, Lula is participating in the presidential election again.*	[*In year 2002,*] [*Lula is participating in presidential election again.*]	[*In year 2002,*] [*Lula is participating in presidential election again.*]
Missed span	Zum Beispiel kann man einen kleinen Text besser lesen.	[*Zum Beispiel kann man besser lesen.*] [*kleinen*]	(only one statement predicted)
(Translation:)	For example, you can read a small text better.	[*For example, you can read better.*] [*small*]	(only one statement predicted)
Excessive tagging	Ein Lurker schreibt selbst keine Artikel in einem Wiki.	[*Lurker schreibt selbst keine Artikel*] [*in Wiki.*]	[*Lurker schreibt selbst*] [*Artikel*] [*in*] [*Wiki.*]
(Translation:)	A lurker does not write articles in a wiki.	[*lurker does not write articles*] [*in wiki.*]	[*lurker writes*] [*articles*] [*in*] [*wiki.*]
Different segmentation	Seit dem Jahr 1983 gibt es Musik Alben von Madonna.	[*Seit Jahr 1983*] [*gibt es Musik Alben*] [*Musik Alben von Madonna*]	[*Seit Jahr 1983*] [*gibt es Musik Alben von*] [Madonna]
(Translation:)	There have been music albums by Madonna since the year 1983.	[*since year 1983.*] [*There have been music albums*] [*music albums by Madonna*]	[*since year 1983.*] [*There have been music albums by*] [Madonna]

Table 1: Example predictions from the development set by our submitted model.

Preliminary experiments with several pre-trained German and multilingual BERT models suggested that the two models `bert-base-german-cased`[4] and `bert-base-multilingual-cased`[5] on the Hugging Face Hub are promising candidates for fine-tuning. We used these two models and performed grid search to optimize hyperparameters such as the number of epochs and batch size. We used span-level F1 score on the development set to determine the best-performing model to submit to the shared task. The code for data preprocessing, model fine-tuning, and the span expansion algorithm is available on GitHub.[6]

Team	Subtask 1 MAE ↓	Subtask 2 chrF ↑	Subtask 2 Jaccard ↑
KlarTextCoder	**0.35**	**0.36**	0.29
CUET_Big_O	0.40	—	—
(Ours)	0.40	0.30	**0.38**
All-1 baseline	0.66	—	—
Random baseline	0.92	0.24	0.27
Conjunction baseline	0.60	0.05	0.04

Table 2: Test set results of all participating teams and baselines provided by the organizers. Best score for each metric is in bold.

4 Results

The best-performing model that emerged from our grid search is based on the multilingually pre-trained BERT with an F1 score of 0.457, clearly outperforming the best German-based BERT model (F1: 0.416). We published our final model on the Hugging Face Hub for reproducibility.[7]

Table 2 shows results on the test set compared to the two other participating teams as well as three baselines provided by the organizers. The all-1 baseline predicts exactly one statement for each sentence. The random baseline predicts a random number of statements between one and three and splits sentences into spans of equal length. The conjunction baseline splits sentences at coordinating conjunctions such as *und* 'and' or *aber* 'but'.

The examples in Table 1 demonstrate that the model is capable of correctly distinguishing between optional and obligatory slots in many cases, but sometimes misses or overgenerates statements. Slight differences in segmentation often arise when the true annotation contains overlapping spans. Overall, the span expansion algorithm appears to work well in most cases.

[4] https://huggingface.co/google-bert/bert-base-german-cased

[5] https://huggingface.co/google-bert/bert-base-multilingual-cased

[6] https://github.com/saeub/statement-segmentation

[7] https://huggingface.co/saeub/bert-stage

5 Discussion

Our approach of only tagging a single token per statement and expanding spans by tracing downward dependencies has several advantages. The abstraction of the task as binary sequence labeling permits a more straightforward implementation compared to previous approaches for discontinuous spans. Separating span identification from span expansion allows a more modular development and granular evalution than end-to-end systems would. For example, it may be possible to adapt the rules in our span expansion algorithm to match the true spans even more closely without retraining the model.

However, the practical benefit of such overly specific optimizations towards exact span expansions is questionable. From an applied perspective, i.e., as part of the writing process, it might be more important to know which semantically central tokens (i.e., span heads) constitute additional statements, as opposed to know whether some preposition or particle belongs to a statement or not.

We would also like to critically put into question the definition of *statement* in terms of valency grammar. Realistic use cases for the statement segmentation task include computer-assisted translation into Easy Language and quality estimation of simplified texts, e.g., by pointing out sentences that should be split into several sentences. However, given the definition in the shared task, splitting the statements into separate sentences substantially changes the meaning in some cases. Consider this sentence from the training set:

(3) *Und man soll auch nicht allein Alkohol trinken.*
'And you shouldn't drink alcohol alone either.'
Statements: [*man soll auch nicht Alkohol trinken.* 'you shouldn't drink alcohol either.'], [*allein* 'alone']

Although syntactically optional, the adverbial *allein* is a crucial modifier of the recommendation not to drink alcohol, limiting its scope to a specific social context. While a grammatical view on *statement* may be easier to define, annotate, and automatically predict, it falls short when considering the semantic content and pragmatic context of the sentences.

6 Conclusion and future work

We presented our submission to the *StaGE* shared task on statement segmentation in German Easy Language. Our approach involves reframing the task as binary sequence labeling and reconstructing spans with a simple rule-based algorithm based on dependency relations. Among three teams, we achieved second place in terms of MAE in Subtask 1 and first place in terms of Jaccard index in Subtask 2.

Our results demonstrate that even generalist BERT models can achieve acceptable performance in emerging tasks in Easy Language. Future work might assess BERT models that are specifically pre-trained on Easy Language data. Anschütz et al. (2023) pre-trained GPT-based models on Easy Language, but to the best of our knowledge, no pre-trained masked language models exist for simplified language varieties. Another line of research could be to investigate alternative, more semantically and pragmatically motivated definitions of *statement*.

Ethical considerations

Easy Language is an important contribution towards more inclusivity and accessibility in society. Research into Easy Language and technology that facilitates the creation of content in Easy Language is essential in this effort. However, it is important to acknowledge that users of Easy Language have diverse needs and requirements, and the effectiveness of guidelines and technologies should always be tested with target users. The *StaGE* shared task and our work focused solely on automatic evaluation metrics, which may not capture this effectiveness well. Therefore, we advise caution when interpreting the results in this paper and encourage future research to investigate the effectiveness of statement segmentation in the creation or evaluation of content in Easy Language.

Acknowledgments

We would like to thank the StaGE organizers for providing the data and organizing the shared task, and the anonymous reviewers for their constructive feedback. We thank the Department of Computational Linguistics at the University of Zurich for providing the computational resources for our experiments.

References

Miriam Anschütz, Joshua Oehms, Thomas Wimmer, Bartłomiej Jezierski, and Georg Groh. 2023. Language models for German text simplification: Overcoming parallel data scarcity through style-specific pre-training. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1147–1158, Toronto, Canada. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

DIN. 2023. Empfehlungen für Deutsche Leichte Sprache (DIN SPEC 33429). Technical report, Deutsches Institut für Normierung e.V.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spacy: Industrial-strength natural language processing in python. *Zenodo*. If you use spaCy, please cite it as below.

Christiane Maaß. 2020. *Easy Language–Plain Language–Easy Language Plus: Balancing comprehensibility and acceptability*. Frank & Timme, Berlin.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Flexible text segmentation with structured multilabel classification. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 987–994, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Alejandro Metke-Jimenez and Sarvnaz Karimi. 2016. Concept identification and normalisation for adverse drug event discovery in medical forums. In *Proceedings of the First International Workshop on Biomedical Data Integration and Discovery (BMDID 2016)*.

Aldrian Obaja Muis and Wei Lu. 2016. Learning to recognize discontiguous entities. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 75–84, Austin, Texas. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Thorben Schomacker, Miriam Anschütz, Regina Stodden, Marina Tropmann-Frick, and Georg Groh. 2024. Overview of the GermEval 2024 Shared Task on Statement Segmentation in German Easy Language (StaGE). In *Proceedings of the GermEval 2024 Shared Task on Statement Segmentation in German Easy Language (StaGE)*, Vienna, Austria. Association for Computational Linguistics.

Buzhou Tang, Jianglu Hu, Xiaolong Wang, and Qingcai Chen. 2018. Recognizing continuous and discontinuous adverse drug reaction mentions from social media using LSTM-CRF. *Wireless Communications and Mobile Computing*, 2018(1):2379208.