ANNOTARES: A Dataset for Extracting Logical Structures from German Statutory Texts

Ronja Schwarz and Jannik Strötgen
Karlsruhe University of Applied Sciences
Moltkestr. 30, 76133 Karlsruhe, Germany
mail@aesz.org
jannik.stroetgen@h-ka.de

Abstract

The automatic structural analysis of legal texts is a cornerstone of legal technology, yet the extraction of their logical components remains a significant challenge. In this paper, we introduce the task of identifying and segmenting legal conditions (*Tatbestand*) and legal consequences (*Rechtsfolge*) within German statutory texts. To support this task, we present ANNOTARES (Annotations of Tatbestand-Rechtsfolge Sequences), a novel dataset comprising German law texts with span-level annotations. Spanning three distinct legal codes, the dataset is designed to evaluate both domain-specific performance and cross-statute generalizability. We benchmark diverse architectural approaches: a rule-based baseline, CRFs, BiLSTMs, BiLSTM-CRF, and modern Transformer-based models, including BERT variants and LLM-based methods. Our results demonstrate that BERT and LLM-based models achieve superior performance in capturing the complex syntactic structures of legal language. We release our dataset to facilitate further research in automated legal reasoning.[1]

1 Introduction

Recent advancements in natural language processing, specifically the rise of large language models (LLMs), have greatly enhanced the capacity for the nuanced structural analysis and automated reasoning required in the legal domain (Siino et al., 2025). However, the precise extraction of the underlying logical structures within statutory texts – specifically the identification of legal conditions (*Tatbestand*) and legal consequences (*Rechtsfolge*) within German law – remains a significant technical challenge. This difficulty largely stems from the fact that German statutory texts are crafted for human interpretation, characterized by highly nested phrasing and implicit legal contexts that complicate automated parsing.

In Germany, the federal government maintains a manual for drafting legislation[2] to unify ministerial drafts and ensure ease of understanding, a uniform structure, and a consistent stylistic approach. Although the first version of the manual was already released in 1991, the principles of the 2024 edition (4th ed.) have not yet been reflected in the vast majority of existing statutory texts. While the manual does not yet aim for machine-executable drafting for new regulations, the suggested uniform syntax and predictable structure is a prerequisite. In addition, recent initiatives (e.g., the Law as Code initiative of the Federal Agency for Disruptive Innovation[3]) propose machine-executable drafting for new regulations. Nevertheless, as of today, the overwhelming majority of existing statutory texts possess complex syntactic structures. Even with the capabilities of modern LLMs, explicitly transforming these texts into structured logical components remains a fundamental prerequisite for reliable automated legal reasoning.

Within the broader LegalNLP domain, significant progress has been made in tasks such as named entity recognition (e.g., Leitner et al. 2019; Au et al. 2022), judgment prediction (e.g., Chalkidis et al. 2019; Feng et al. 2022), and document summarization (e.g., Kanapala et al. 2019; Jain et al. 2024). However, these tasks usually treat legal text as a collection of entities or a sequence of labels without capturing the underlying logic defining a legal norm. Transforming statutory prose into a form suitable for automated reasoning requires a more fine-granular approach that segments text into its logical components: legal prerequisite conditions and the resulting legal consequences.

In this paper, we introduce a novel sequence tagging task for the legal domain to extract logical components from statutory text: the identifica-

[1] https://github.com/wilhelmines/annotares

[2] https://hdr4.bmj.de/

[3] https://www.sprind.org/en/actions/strategic-projects/law-as-code

tion and segmentation of legal conditions (*Tatbestand*) and legal consequences (*Rechtsfolge*). For this, we introduce the ANNOTARES dataset (Annotations of Tatbestand-Rechtsfolge Sequences), which spans three distinct legal codes: the Federal Data Protection Act (*Bundesdatenschutzgesetz*, BDSG), the Federal Education and Training Assistance Act (*Bundesausbildungsförderungsgesetz*, BAföG), and the Federal Building Code (*Baugesetzbuch*, BauGB).

ANNOTARES includes a primary corpus of more than 400 annotated sentences from the BDSG, split into training and test sets, complemented by 50 annotated sentences each from the BAföG and BauGB to serve as out-of-domain test sets for assessing cross-statute generalizability. The dataset contains more than 21,500 annotated tokens with approximately 43%, 44% and 13% of them being annotated as legal conditions (*Tatbestand*), legal consequence (*Rechtsfolge*) and none, respectively. Despite the low proportion of the none class, a token-level inter-annotator agreement analysis yielded a Krippendorff's alpha of 0.89, indicating high annotation quality and robust boundary detection.

As a further main contribution of this paper, we benchmark six diverse architectural approaches: a rule-based baseline, CRFs, BiLSTMs, BiLSTM-CRF, and modern Transformer-based models, including BERT variants and LLM-based methods. Our results demonstrate that BERT and LLM-based models achieve superior performance, successfully capturing the complex syntactic structures inherent in legal language.

The remainder of the paper is structured as follows: After surveying related work in Section 2, we define the task and describe the annotation guidelines in Section 3. The dataset annotation process and detailed corpus statistics are presented in Section 4 accompanied with detailed dataset statistics. Finally, Section 6 describes our experimental setup and evaluation results, before we conclude and outline future research directions in Section 7.

2 Related Work

Although the legal domain is a highly specialized field, it supports an active research community. Several works addressed legal document summarization (e.g., Kanapala et al. 2019; Jain et al. 2024), legal question answering (e.g., Askari et al. 2024; Büttner and Habernal 2024; Louis et al. 2024), and legal judgment prediction (e.g., Chalkidis et al. 2019; Feng et al. 2022). Furthermore, various works addressed text classification for various types of legal text documents (e.g., Chalkidis et al. 2021; Wang et al. 2022), including patents (e.g., Pujari et al. 2021).

Glaser et al. (2021) addressed the task of sentence boundary detection in German legal documents demonstrating that tackling NLP tasks in the legal domain requires domain adaptation similar to many other domains in other low-resource scenarios (Hedderich et al., 2021).

More closely related to our work, there are also some works targeting sequence tagging tasks. Most prominently, named entity recognition (NER) in the legal domain was addressed in several languages (Pais et al., 2021; Au et al., 2022; Smădu et al., 2022). Leitner et al. (2019, 2020) addressed fine-grained NER and developed a respective dataset covering German legal documents. They used Bi-LSTMs and CRFs to perform the sequence tagging – architectures that we also include in our benchmarking experiments. Furthermore, the development of domain-specific language models, such as LEGAL-BERT (Chalkidis et al., 2020), has demonstrated that pre-training on large-scale legal corpora can significantly improve performance on downstream tasks compared to general-purpose models. In addition, research competitions such as COLIEE (Goebel et al., 2024) have fostered innovation in legal information extraction and entailment.

There are also recent surveys covering the LegalNLP domain. Ariai et al. (2025) provide a broad overview of natural language processing in the legal domain. They discuss various tasks, existing datasets as well as used models to tackle the discussed NLP tasks in the legal domain. Siino et al. (2025) survey recent research focussing on LLM-based approaches.

However, to the best of our knowledge, no existing work targets the extraction of logical components in statutory text. In contrast, we define the novel task of identifying and segmenting legal prerequisite conditions (*Tatbestand*) and the resulting legal consequences (*Rechtsfolge*). Consequently, the manual for drafting legislation (*Handbuch der Rechtsförmlichkeit*) released by the German Federal Ministry of Justice (Bundesministerium der Justiz, 2024) serves as a foundational resource for our study. This manual provides the structural and syntactic guidelines and offers standardized examples of the uniform syntactic structures we aim to

extract. Based on them, we defined our annotation guidelines for the novel task of extracting legal conditions and legal consequences in German statutory texts.

3 Annotation Guidelines

Task Definition. We define the extraction of logical components from statutory texts as a sequence tagging task. Each token in a sentence is assigned one of three labels: legal conditions (*Tatbestand*) or legal consequences (*Rechtsfolge*) or as *none* (the background class).

In the civil law tradition, legal conditions (*Tatbestand*) and legal consequences (*Rechtsfolge*) constitute the foundational structure of legal propositions. Dividing a sentence into these functional parts allows automated systems (e.g., legal knowledge graphs, retrieval-augmented generation (RAG) pipelines, and neuro-symbolic frameworks) to parse the underlying logic of a norm. By explicitly segmenting these structures, downstream applications can focus on semantic interpretation and legal reasoning without the added burden of structural disambiguation. Our approach assumes two core principles derived from the manual for drafting legislation (Bundesministerium der Justiz, 2024): (1) *Atomicity*, where each sentence contains exactly one legal statement; and (2) *Completeness*, where every sentence is either a formal definition or a combination of legal conditions and legal consequences.

Annotation Guidelines. The guidelines were developed to facilitate the annotation of German statutory texts based on identifiable linguistic patterns. This ensures that even non-domain experts can perform the task in a uniform, reproducible manner.

The framework defines rules for assigning the classes (i) *Tatbestand* (legal conditions), (ii) *Rechtsfolge* (legal consequence), or (iii) *none* (i.e., the "outside" class). A key structural constraint of our guidelines is the interdependency of the logical labels: if a sentence contains a *Tatbestand* (legal conditions), it must also contain a *Rechtsfolge* (legal consequence), and vice versa. Sentences that do not follow this conditional structure (e.g., purely introductory text) are labeled entirely as *none*.

Representative examples of the annotation schema are depicted in Figure 1. Example 1 illustrates a standard linear structure, where the initial segment is identified as *Tatbestand* (legal conditions) followed by the *Rechtsfolge* (legal consequence). Example 2 demonstrates that this logical order might be inverted, while Example 3 showcases a more complex syntactic structure containing multiple, interleaved spans of both conditions (*Tatbestand*) and consequences (*Rechtsfolge*). Finally, Example 4 illustrates the *none* class. In this instance, the sentence provides a formal definition that states a legal fact without prescribing a specific normative consequence, thus falling outside the scope of our primary logical extraction task.

Example 1 (§ 43 (2) BDSG):
Die Ordnungswidrigkeit kann mit einer Geldbuße bis zu fünfzigtausend Euro geahndet werden.
(An administrative offence may be punished by a fine of up to fifty thousand euros.)

Example 2 (§6 (4) BDSG):
Die Kündigung des Arbeitsverhältnisses ist unzulässig, es sei denn, dass Tatsachen vorliegen, welche die öffentliche Stelle zur Kündigung aus wichtigem Grund ohne Einhaltung einer Kündigungsfrist berechtigen.
(The data protection officer's employment shall not be terminated unless there are facts which give the public body just cause to terminate without notice.)

Example 3 (§79 (2) BDSG):
Sie [die Dokumentation] ist der oder dem Bundesbeauftragten auf Anforderung zur Verfügung zu stellen.
(It [the documentation] shall be provided to the Federal Commissioner on request.)

Example 4 (§8 (1) BDSG):
Der Dienstsitz ist Bonn.
(It is located in Bonn.)

Figure 1: Representative sample sentence annotations from the BDSG. Spans identified as *Tatbestand* (legal conditions) are highlighted in yellow, whiles *Rechtsfolge* (legal consequence) spans are highlighted in blue. Unhighlighted text represents the *none* class. English translations are sourced from the official federal version of the BDSG (`https://www.gesetze-im-internet.de/englisch_bdsg/englisch_bdsg.html`).

Tatbestand. Legal conditions (i.e., *Tatbestand*) describe the prerequisites that must be satisfied for the associated legal consequences to take effect. The manual for drafting legislation (Bundesministerium der Justiz, 2024) recommends a clear structure in sentences to allow for a simple identification of the part that is the constituent element. It recommends using conditional clauses to make these elements easily identifiable. In particular, it suggests keywords such as *wenn*, *falls*, and *sofern* (corresponding to conditionals such as "if", "in case", or "provided that").

However, legal conditions are not always introduced by explicit conjunctions. They may appear

without a conditional clause as in Example 1, Figure 1: **Die Ordnungswidrigkeit** [Tatbestand] kann mit einer Geldbuße [...] geahndet werden" (**An administrative offense** [Legal Condition] may be punished by a fine [...]").

Rechtsfolge. The legal consequence (i.e., *Rechtsfolge*) specifies the normative result triggered when the corresponding legal conditions are met. In German statutory language, consequences are frequently characterized by the use of modal verbs that define the degree of obligation: *kann* (permissive/discretionary), *soll* (expected/guided discretion), and *muss* (mandatory).

To resolve potential ambiguities, the guidelines provide specific rules for mentions of persons and institutions. When an actor is part of the situational prerequisite, they are labeled as *Tatbestand* (legal conditions). If the actor is primarily the subject or object of the resulting legal action, they are included in the *Rechtsfolge* (legal consequence) span.

4 Dataset Creation and Dataset Statistics

Annotation Procedure. The ANNOTARES corpus was annotated by a team of six individuals, including the lead author and five additional annotators. The lead author annotated the entire corpus to provide a consistent expert baseline. Each sentence was also assigned to at least one of the other five annotators, each of whom completed a minimum of 80 sentences. This overlapping assignment strategy ensured that every sentence in the dataset had at least two independent annotations for verification. To maintain high data quality, annotators were provided with an "unsure" flag for ambiguous cases. Any sentence marked with this flag was excluded from both the inter-annotator agreement calculation and the final dataset, resulting in the removal of seven sentences.

Data Preprocessing. Legal texts were sourced in XML format from the official federal repository.[4] These files contain rich metadata, including legislative years and hierarchical document positions. Paragraphs were extracted as individual elements, while sentence segmentation was performed using a hybrid approach: initial automated sentence splitting followed by rigorous manual correction. Manual intervention was necessary due to the characteristic length and complex punctuation of German legal prose, which often lead to errors in standard sentence splitters.

The annotation campaign utilized a custom-developed, Python-based annotation tool. While existing platforms such as Inception (Klie et al., 2018) or Doccano (Nakayama et al., 2018) offer extensive features, they often present a steep learning curve for non-experts. Our streamlined interface was designed to reduce cognitive load and simplify data management by eliminating the need for complex file-format conversions.

Dataset Coverage and Splits. ANNOTARES comprises sentences from three federal German laws: the Federal Data Protection Act (*Bundesdatenschutzgesetz*, *BDSG*), the Federal Education and Training Assistance Act (*Bundesausbildungsförderungsgesetz*, *BAföG*), and the Federal Building Code (*Baugesetzbuch*, *BauGB*). The *BDSG* was annotated in its entirety and serves as the primary corpus for training and evaluation. It was randomly split into training, validation, and test sets containing 351, 43, and 45 sentences, respectively. The *BAföG* and *BauGB* were sampled (50 sentences each) to serve as distinct out-of-domain test sets to evaluate cross-statute generalizability. These samples were selected based on structural parity, specifically matching the word count and frequency distributions observed in the primary *BDSG* corpus.

Dataset Statistics. As shown in Table 1, the three statutes exhibit similar structural patterns, with mean sentence lengths ranging from 38.1 to 42.4 tokens. The *BDSG* displays a higher proportion of the *none* class (+8 percentage points) compared to the sampled sets. This discrepancy is likely due to the inclusion of the complete *BDSG* text, which encompasses introductory and administrative clauses that lack a conditional structure, whereas the sampled sets targeted more dense normative content.

Linguistic Preprocessing. To provide sophisticated features for traditional machine learning models (e.g., CRFs), we enriched the dataset with dependency parsing information in the CoNLL-U format (Nivre et al., 2016). For this, we evaluated two parsers: spaCy (`de_core_news_sm`) (Honnibal et al., 2020) and Stanza (`UD-German_GSD`) (Qi et al., 2020). Following a preliminary comparison in which Stanza demonstrated a slight performance lead, we selected the Stanza model to generate the dependency features used in our benchmarking experiments.

[4] https://www.gesetze-im-internet.de

Law	Splits	Sentences	Tokens	Coverage of tokens (%) Rechtsfolge (leg. consequence)	Tatbestand (leg. conditions)	None (outside)
BDSG	Train, Val, Test	439	17,524	44.3	40.6	15.1
BAföG	Test	50	2,121	38.0	57.6	4.4
BauGB	Test	50	1,905	46.3	46.6	7.1
overall		539	21,550	43.8	42.8	13.4

Table 1: Statistics for the three splits of the ANNOTARES dataset.

Dataset	Annotators	α
BAföG	All (2)	0.897
BauGB	All (2)	0.926
BDSG	All (6)	0.893
	00 & 01	0.896
	00 & 02	0.858
	00 & 03	0.826
	00 & 04	0.942
	00 & 05	0.923

Table 2: Inter-annotator agreement analysis on the ANNOTARES corpus based on Krippendorff's α. Individual agreement scores between the expert and each contributing annotator per source.

Inter-Annotator Agreement. We employed Krippendorff's Alpha (α) to measure agreement, as it robustly handles nominal labels and varying annotation counts per token. In our setup, the lead author (Annotator 00) served as the expert reference, having annotated the entire corpus. Individual agreement scores were calculated between the expert and each contributing annotator. The datasets from the *BAföG* and the *BauGB* were annotated by two persons each, while the *BDSG* includes the results of six different annotators.

As shown in Table 2, the datasets achieved high reliability scores: α=0.897 for *BAföG*, α=0.926 for *BauGB*, and a cumulative α=0.893 for the *BDSG*. Following the initial campaign, an adjudication step was performed by the lead author to harmonize the final labels. In 84.4% of cases, the annotations were either identical or resolved through deterministic corrections of trivial errors (e.g., adjusting misplaced punctuation at span boundaries). The remaining disputes were resolved based on the expert's deeper task understanding, with the final labels favoring the expert in 12.7% of instances and the contributing annotator in 2.9%.

5 Experimental Setup

We evaluate six distinct architectural approaches for the logical segmentation of statutory text. These models were selected to represent a spectrum of complexity, computational requirements, and interpretability.

As a baseline for simplicity and human interpretability, we implemented a **rule-based approach**. We further included a **Conditional Random Field (CRF)** model as a representative for a probabilistic model (Lafferty et al., 2001) to represent traditional probabilistic sequence modeling, which has historically been the standard for linguistic structure extraction.

To capture long-range dependencies, we implemented **Bi-directional Long Short-Term Memory (BiLSTM)** networks and a combined **BiLSTM-CRF** architecture. The latter has consistently demonstrated high performance in named entity recognition and related sequence tagging tasks Lample et al. (2016).

We further benchmarked several variants of the **BERT** architecture (Devlin et al., 2019). Our selection includes general-purpose models such as *mBERT* and *DistilBERT* (Sanh et al., 2019), as well as domain-specific variants like *LEGAL-BERT* (Chalkidis et al., 2020) and *BERT-LER*. These models utilize the Transformer's encoder block to generate nuanced, context-aware representations (Vaswani et al., 2017).

Finally, we evaluated the zero-shot and few-shot capabilities of modern **Large Language Models (LLMs)**, specifically *Claude 3 Haiku* and *Claude 3 Opus* (Anthropic, 2024). Unlike the encoder-only BERT models, these are decoder-only architectures (Radford et al., 2018) capable of following complex instructions through natural language prompts.

Detailed hyperparameter configurations for all evaluated models are provided in the appendix (Appendix A).

Rules. We implemented a sequential three-stage rule-based system. The first rule matches verb-first (V1) sentence structures, which is a common indicator of conditional clauses in German legal prose. The second rule utilizes a set of keywords (e.g., *wenn*, *falls*) as indicators for a *Tatbestand* (legal conditions). These sentences are split at the primary comma, with the remaining segment labeled as *Rechtsfolge* (legal consequence). Finally, a catch-all rule assigns the *none* label to any sentence not captured by the preceding rules.

CRF. The CRF was implemented using the `python-crfsuite` library with default hyperparameters. Feature engineering included token-level attributes (text, casing, numerical flags), part-of-speech (POS) tags, and dependency labels. To capture local context, we utilized n-grams and a sliding window that incorporates the POS and dependency tags for the immediate neighbors tokens.

BiLSTM and BiLSTM-CRF. Both neural architectures utilize pre-trained `fastText` embeddings (Grave et al., 2018) trained on German Common Crawl and Wikipedia data. These embeddings are concatenated with learned embeddings for POS and dependency tags before being processed by a Bi-directional LSTM. In the standard BiLSTM, a linear layer with a softmax activation provides the class probabilities. The BiLSTM-CRF variant extends this by passing the LSTM outputs into a CRF layer to model label dependencies. Both models were implemented using `PyTorch`.

BERT Variants. We evaluated twelve German-centric and multilingual models sourced from the Hugging Face Hub. Following a preliminary screening, the four top-performing models were selected for full evaluation: *BERT-LER*, *DistilBERT*, *LEGAL-BERT*, and *mBERT*.[5] Our architecture concatenates the BERT hidden states with POS and dependency tag embeddings, followed by a linear classification layer. Models were trained across three random seeds to ensure statistical robustness.

Large Language Models (LLMs). We evaluated *Claude Opus 4.6* and *Claude Haiku 4.5* using two distinct prompting strategies. The first prompt was manually created with general prompt suggestions provided by ChatGPT, while the second prompt was created by Claude Sonnet 4.5 with the goal to adapt it for Claude Opus. Prompt 1 serves as a textual representation of the annotation guidelines, including class definitions, mandatory structural rules, and few-shot examples. Prompt 2 is an optimized version designed to leverage the models' reasoning capabilities. It incorporates structured POS and dependency tag mappings and extended examples to guide the segmentation process. Both models were tested in zero-shot and few-shot configurations to assess their inherent understanding of legal logic. The full text of both prompts is available in Appendix B.

[5] mrm8488/bert-base-german-finetuned-ler (BERT-LER), dbmdz/distilbert-base-german-europeana-cased (DistilBERT), nlpaueb/legal-bert-base-uncased (LEGAL-BERT) , google-bert/bert-base-multilingual-cased (mBERT).

Method	Accuracy	mF1
Rules	0.161	0.197
CRF	0.711	0.608
BiLSTM	0.529	0.357
BiLSTM + CRF	0.572	0.480
BERT-LER	<u>0.842</u>	<u>0.782</u>
DistilBERT	0.784	0.713
LEGAL-BERT	0.787	0.740
mBERT	**0.854**	**0.798**
LLM Prompt 1 + Haiku	0.826	0.713
LLM Prompt 1 + Opus	0.820	0.735
LLM Prompt 2 + Haiku	0.703	0.606
LLM Prompt 2 + Opus	0.678	0.619

Table 3: Comparison of the approaches on the BDSG test split of the ANNOTARES data set. Best scores in bold, second best underlined.

6 Evaluation

In this section, we provide a comprehensive evaluation of the automated models against the human-annotated gold standard. We structure our analysis into four parts: first, we assess in-domain token-level performance on the primary BDSG test split. We then conduct a detailed error analysis and evaluate precise boundary detection using exact span matches. Third, we test the models' cross-statute generalization capabilities on the out-of-domain BAföG and BauGB test sets. Finally, we present an ablation study to quantify the contribution of structural linguistic features, specifically part-of-speech and dependency tags, to the overall performance.

All methods were evaluated across the three ANNOTARES test sets. We report Macro F1 score ($mF1$) as our primary metric to account for the uneven distribution of tokens across the three classes (cf. Table 1), alongside accuracy for completeness.

In-Domain Experiments. Table 3 summarizes the performance of all evaluated methods on the *BDSG* test split. The results highlight a significant performance gap between traditional baselines and modern Transformer-based architectures. The manual rule-based system performs poorly ($mF1 = 0.197$), largely due to its inability to handle the syntactic complexity of legal prose, resulting in an over-reliance on the *none* class (see Figure 2 in the appendix). Similarly, the BiLSTM architectures struggled ($mF1 < 0.50$), likely due to the limited size of the training data and no applied transfer learning.

All other methods show promising results for the novel task of identifying and segmenting legal conditions (*Tatbestand*) and legal consequences (*Rechtsfolge*) within German statutory texts. mBERT achieved the highest overall performance with an accuracy of 85.4% and an $mF1$ score of 0.798. This suggests that multilingual pretraining provides a robust foundation for capturing the nuances of German legal logic. LLMs (Haiku and Opus) utilizing the guideline-based prompt 1 demonstrated competitive results, nearly matching their performance of domain-specific models like *BERT-LER*. However, prompt 2 (despite incorporating explicit linguistic features such as dependency tags) yielded lower $mF1$ scores. This suggests that increasing prompt complexity may introduce noise that detracts from the model's instruction-following capabilities.

Error Analysis and Span Matching. Confusion matrices (figures 2 to 5 in the appendix) for all approaches reveal a systematic bias: most models favor the *Rechtsfolge* (legal consequence) over the other classes. Most models exhibit a consistent performance hierarchy where *Rechtsfolge* (legal consequence) is most accurately identified, followed by *Tatbestand* (legal conditions), and finally the *none* class. A notable exception is the BiLSTM model, which failed to predict the *none* class entirely.

To gain deeper insights, we evaluated exact span matches (Table 4). A span is only considered correct if its boundaries match the gold standard exactly without gaps or shifts. We specifically highlight performance on the *none* class, as it serves as a proxy for how well a model internalized the underlying structural rules of the text (e.g., distinguishing functional legal rules from background definitions). While the BERT-based models remained consistent (0.5 ± 0.1 match rate for the *none* class), *Opus* with Prompt 2 exhibited a unique error profile: it reversed the common class-priority and achieved the highest exact span match for the *none* class (0.875), despite lower overall token accuracy.

The discrepancy between token accuracy and lower span-match scores for neural models underscores the difficulty of the task. This is most dramatically illustrated by the BiLSTM-CRF, which achieved a 0.772 token accuracy for *Rechtsfolge* but completely failed to predict a single exact span for *Rechtsfolge* and *Tatbestand* (0.000). Even if a model correctly identifies the core of a legal condition, failing to precisely identify the boundary results in a total span failure. Finally, with the notable exception of the *Opus* model (in particular with Prompt 2), we observed that the systems generally struggled to detect or enforce the underlying logical rule of the corpus, i.e., the fundamental incompatibility of *none* labels with *Tatbestand* (legal conditions) / *Rechtsfolge* (legal consequence) within the same normative sentence.

Cross-Statute Generalization. To evaluate the robustness of the models across different legal domains, we conducted a generalization study using the *BAföG* and *BauGB* test sets. For this experiment, we utilized the best-performing configurations from our in-domain tests: the *mBERT* model and the top-performing seeds for the BiLSTM variants. We excluded the LLMs from this specific comparison to focus on the transferability of the trained sequence taggers.

As shown in Table 5, the rule-based baseline demonstrated slightly higher performance on the *BAföG* and *BauGB* test sets than on the *BDSG*. This suggests that the sampled sentences from these statutes happen to align more closely with the specific V1 and keyword-based structures defined in our rules, though overall performance remains insufficient for practical use.

The CRF yielded mixed results. While it achieved its highest accuracy on the in-domain *BDSG* set (0.711), its $mF1$ score was significantly higher on the *BauGB* set (0.648). The BiLSTM variants showed a slight in-domain bias, with accuracy drops of up to 0.059 when moving to out-of-domain data. However, the BiLSTM-CRF showed inconsistent generalization behavior, achieving its highest $mF1$ on the *BAföG* set (0.566) despite a lower accuracy, indicating a high variance in how the CRF layer handles the label transitions in different legislative contexts.

Method	Token Accuracy			Exact Span Match		
	Rechtsfolge	Tatbestand	None	Rechtsfolge	Tatbestand	None
Rules	0.029	0.184	1.000	0.061	0.047	1.000
CRF	0.770	0.675	0.449	0.367	0.465	0.500
BiLSTM	0.706	0.368	0.000	0.042	0.032	0.000
BiLSTM + CRF	0.772	0.408	0.087	0.000	0.000	0.042
BERT-LER	0.893	0.793	0.542	0.389	0.595	0.458
DistilBERT	0.885	0.704	0.504	0.306	0.444	0.500
LEGAL-BERT	0.860	0.719	0.509	0.368	0.460	0.417
mBERT	0.876	0.850	0.589	0.438	0.579	0.542
LLM Prompt 1 + Haiku	0.903	0.796	0.409	0.408	0.395	0.375
LLM Prompt 1 + Opus	0.861	0.786	0.676	0.469	0.581	0.750
LLM Prompt 2 + Haiku	0.760	0.652	0.523	0.163	0.209	0.625
LLM Prompt 2 + Opus	0.662	0.683	0.761	0.306	0.186	0.875

Table 4: Token accuracy and exact span match per class (BDSG).

	Rules		CRF		BiLSTM		BiLSTM + CRF		mBERT	
	Acc	mF1	Acc	mF1	Acc	mF1	Acc	mF1	Acc	mF1
BAföG	0.231	0.235	0.595	0.465	0.470	0.320	0.517	0.566	0.854	0.868
BauGB	0.315	0.313	0.685	0.648	0.490	0.338	0.455	0.455	0.843	0.812
BDSG	0.161	0.197	0.711	0.608	0.529	0.357	0.572	0.480	0.854	0.798

Table 5: Results of the cross-statute generalization study showing the scores on the three ANNOTARES test sets.

The mBERT model exhibited the most impressive generalization capabilities. It maintained a near-constant accuracy across all three statutes (ranging from 0.843 to 0.854). Remarkably, the $mF1$ scores for the out-of-domain sets (*BAföG*: 0.868; *BauGB*: 0.812) actually exceeded the in-domain score (0.798). This suggests that the model has successfully captured universal structural patterns of German statutory law that transcend specific administrative domains.

The performance fluctuations across the more traditional models (CRF and BiLSTM) further reinforce our earlier observation: a training set of 351 sentences appears insufficient for these architectures to reach a stable state of generalization. In contrast, the Transformer-based approach leverages its extensive pre-training to overcome these data limitations, proving highly effective for cross-statute logical segmentation.

Ablation Study. To quantify the contribution of structural linguistic information to the segmentation task, we conducted an ablation study by removing part-of-speech (POS) and dependency (DEP) tags from our top-performing models. All ablated models were evaluated on the *BDSG* test set using the same seed and architecture configurations as the primary experiments. Note that our ablated BiLSTM-CRF follows the standard architecture proposed by Lample et al. (2016), utilizing only token-level embeddings.

Model	POS/DEP	Acc	mF1
CRF	yes	0.711	0.608
CRF	no	0.691	0.589
BiLSTM	yes	0.529	0.357
BiLSTM	no	0.534	0.259
BiLSTM + CRF	yes	0.572	0.480
BiLSTM + CRF	no	0.582	0.292
mBERT	yes	0.854	0.798
mBERT	no	0.757	0.679

Table 6: Results of the ablation study.

As shown in Table 6, the inclusion of POS and DEP features significantly enhances performance across nearly all architectures, particularly in terms of $mF1$. The Transformer (mBERT) model exhibited the most dramatic sensitivity to these features. Removing linguistic tags caused the $mF1$ to drop from 0.798 to 0.679 and accuracy to fall by nearly 10 percentage points. This indicates that while mBERT has strong internal representations, explicit syntactic features are crucial for anchoring legal logic to specific grammatical structures.

The BiLSTM variants presented a unique behavior. While their accuracy slightly increased or remained stable upon removing linguistic features, their $mF1$ scores dropped severely (e.g., from 0.480 to 0.292 for BiLSTM-CRF) – highlighting that token accuracy should not be used as the only metric for this task. An inspection of the confusion matrices (figures 6 to 8 in the appendix) reveals that without linguistic context, these models essentially collapse toward the majority class (*Rechtsfolge*). The ablated BiLSTM, for instance, showed no awareness of the *none* class and lost significant precision in identifying *Tatbestand* boundaries.

Similarly, the CRF model's performance decreased across both metrics, though it remained more robust than the neural variants. Across all models, the removal of POS and DEP tags exacerbated the systemic bias toward the majority class. This suggests that the structural language information we provide serves as a necessary counterweight to the statistical bias inherent in the dataset, allowing models to identify the logical conditions (*Tatbestand*) that are often defined by specific syntactic dependencies rather than just vocabulary.

7 Conclusions and Future Work

In this paper, we introduced ANNOTARES, a novel dataset for the automated logical segmentation of German statutory text into *Tatbestand* (conditions) and *Rechtsfolge* (consequences). Our benchmarking of six architectural approaches demonstrates that while traditional and neural baselines struggle with data scarcity, Transformer-based models (specifically mBERT) provide a robust foundation for this task. Notably, although LLMs showed lower overall token-level performance, they outperformed all other models in exact span matching for the background (*none*) class.

The results confirm that structural linguistic features (POS and dependency tags) are essential for capturing legal logic in traditional and fine-tuned neural architectures, as their removal led to a significant performance collapse. In contrast, explicitly injecting these structural features into LLM prompts (Prompt 2) degraded overall token-level performance, suggesting that the added prompt complexity introduces noise rather than aiding extraction.

Building on the ANNOTARES foundation, future work should focus on: (i) Dataset Expansion: Incorporating EU and DACH-region statutes to enhance cross-jurisdictional generalizability; (ii) Two-Stage Modeling: Developing pipeline architectures that first filter normative vs. non-normative sentences before performing token-level extraction; and (iii) Domain-Specific Tooling: Evaluating the performance of dependency parsers fine-tuned specifically on legal corpora to replace general-domain baselines.

By providing ANNOTARES and this initial benchmark, we aim to facilitate new research into logic-aware NLP for the legal domain.

Acknowledgments

The authors want to thank Ben Dondelinger, Frederik Höft, Hannah Henze, Katharina Wittemann, Marius Höll, Markus Roos and Tanja Schwarz for their support during the annotation campaign. In addition, we thank the anonymous reviewers for their helpful comments to further improve the paper.

Generative AI Assistance Declaration. During the preparation of this work, the authors used generative AI tools to rephrase and proofread text content. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

Anthropic. 2024. The Claude 3 model family: Opus, Sonnet, Haiku. Technical report, Anthropic.

Farid Ariai, Joel Mackenzie, and Gianluca Demartini. 2025. Natural language processing for the legal domain: A survey of tasks, datasets, models, and challenges. *ACM Comput. Surv.*, 58(6).

Arian Askari, Zihui Yang, Zhaochun Ren, and Suzan Verberne. 2024. Answer retrieval in legal community question answering. In *Advances in Information Retrieval: 46th European Conference on Information Retrieval, ECIR 2024, Glasgow, UK, March 24–28, 2024, Proceedings, Part III*, page 477–485, Berlin, Heidelberg. Springer-Verlag.

Ting Wai Terence Au, Vasileios Lampos, and Ingemar Cox. 2022. E-NER — an annotated named entity recognition corpus of legal text. In *Proceedings of the Natural Legal Language Processing Workshop 2022*, pages 246–255, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Bundesministerium der Justiz. 2024. *Handbuch der Rechtsförmlichkeit*, 4 edition. Bundesministerium der Justiz. Empfehlungen des Bundesministeriums

der Justiz für die rechtsförmliche Gestaltung von Gesetzen und Rechtsverordnungen nach § 42 Absatz 4 und § 62 Absatz 2 der Gemeinsamen Geschäftsordnung der Bundesministerien. Bekanntgemacht im BAnz AT 31.10.2024.

Marius Büttner and Ivan Habernal. 2024. Answering legal questions from laymen in German civil law system. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2015–2027, St. Julian's, Malta. Association for Computational Linguistics.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural legal judgment prediction in English. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4317–4323, Florence, Italy. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, and Ion Androutsopoulos. 2021. MultiEURLEX - a multi-lingual and multi-label legal document classification dataset for zero-shot cross-lingual transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6974–6996, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online. 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.

Yi Feng, Chuanyi Li, and Vincent Ng. 2022. Legal judgment prediction via event extraction with constraints. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 648–664, Dublin, Ireland. Association for Computational Linguistics.

Ingo Glaser, Sebastian Moser, and Florian Matthes. 2021. Sentence boundary detection in german legal documents. In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART*, pages 812–821. INSTICC, SciTePress.

Randy Goebel, Yoshinobu Kano, Mi-Young Kim, Juliano Rabelo, Ken Satoh, and Masaharu Yoshioka. 2024. Overview and discussion of the competition on legal information, extraction/entailment (COLIEE) 2023. *The Review of Socionetwork Strategies*, 18(1):27–47.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Deepali Jain, Malaya Dutta Borah, and Anupam Biswas. 2024. A sentence is known by the company it keeps: Improving legal document summarization using deep clustering. *Artificial Intelligence and Law*, 32(1):165–200.

Ambedkar Kanapala, Sukomal Pal, and Rajendra Pamula. 2019. Text summarization from legal documents: a survey. *Artificial Intelligence Review*, 51(3):371–402.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, Santa Fe, New Mexico. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Elena Leitner, Georg Rehm, and Julian Moreno-Schneider. 2019. Fine-grained named entity recognition in legal documents. In *Semantic Systems. The Power of AI and Knowledge Graphs. SEMANTiCS*

2019, pages 272–287, Cham. Springer International Publishing.

Elena Leitner, Georg Rehm, and Julian Moreno-Schneider. 2020. A dataset of German legal documents for named entity recognition. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4478–4485, Marseille, France. European Language Resources Association.

Antoine Louis, Gijs van Dijck, and Gerasimos Spanakis. 2024. Interpretable long-form legal question answering with retrieval-augmented large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(20):22266–22275.

Hiroki Nakayama, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi, and Xu Liang. 2018. doccano: Text annotation tool for human. Software available from https://github.com/doccano/doccano.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Vasile Pais, Maria Mitrofan, Carol Luca Gasan, Vlad Coneschi, and Alexandru Ianov. 2021. Named entity recognition in the Romanian legal domain. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 9–18, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Subhash Chandra Pujari, Annemarie Friedrich, and Jannik Strötgen. 2021. A multi-task approach to neural multi-label hierarchical patent classification using transformers. In *Advances in Information Retrieval*, pages 513–528, Cham. Springer International Publishing.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Technical report, OpenAI.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Marco Siino, Mariana Falco, Daniele Croce, and Paolo Rosso. 2025. Exploring LLMs applications in law: A literature review on current legal NLP approaches. *IEEE Access*, 13:18253–18276.

Răzvan-Alexandru Smădu, Ion-Robert Dinică, Andrei-Marius Avram, Dumitru-Clementin Cercel, Florin Pop, and Mihaela-Claudia Cercel. 2022. Legal named entity recognition with multi-task domain adaptation. In *Proceedings of the Natural Legal Language Processing Workshop 2022*, pages 305–321, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Qiqi Wang, Kaiqi Zhao, Robert Amor, Benjamin Liu, and Ruofan Wang. 2022. D2GCLF: Document-to-graph classifier for legal document classification. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2208–2221, Seattle, United States. Association for Computational Linguistics.

A Hyperparameter

No extensive hyperparameter search has been carried out. The CRF approach uses the default settings by the pycrfsuite package. The features provided to the model are the word/token in lower case form, the ending of previous word, the beginning of following word, is it completely capitalized, is the first letter capitalized, is it a digit, its POS- and dependency-tag. For the neighboring words the features are the lower case word/token, is the first letter capitalized, its POS- and dependency-tag.

The BiLSTM and BiLSTM-CRF models are based on the paper by Lample et al. (2016). The implementation uses the pytorch library. From seeds 1-3 seed 3 was selected. The BiLSTM was trained for 11 epochs and the BiLSTM+CRF for 37 epochs. The epochs were selected by a patience of 10 and a max epoch count of 100. The initial learning rate is 0.01 with an additional learning rate decay of 0.05 using the formula:

$$\text{Current learning rate} = \frac{0.01}{1 + (0.05 \times \text{epoch})}$$

The selected optimizer is SGD and a clip of 5 was used. The pretrained word embeddings are cc.de.300.vec (Grave et al., 2018) with a word embedding dimension of 100. The model uses a bidirectional LSTM with 100 hidden dimensions and a bidirectional LSTM with 25 hidden dimensions

for the char embeddings. Two additional embedding layers are used for the POS- and dependency-tags with 25 dimensions each. A dropout of 0.5 is applied between the embeddings and the final LSTM layer. The classification is done with a linear classification layer. The CRF variant extends the BiLSTM with a CRF layer using the torchcrf package.

The transformer approach combines a pretrained BERT model with two custom embeddings for POS- and dependency-tags. The model was trained for 20 epochs with a patience of 10 and a max epoch count of 100. Based on hardware constraints the batch size of 8 was selected. The seeds 1-3 were trained and seed 1 was selected. After the concatenation of the three embedding layers a dropout of 0.1 was applied. The dropout layer feeds into a linear layer for the classification step. The optimizer is AdamW. The custom embeddings have 64 dimensions each and an initial learning rate of 0.001. The learning rate for the BERT model starts with 0.00002. The scheduler ReduceLROnPlateau was chosen for the learning rate. The settings are mode max, factor 0.5 and patience 2.

B Prompts

The Claude Opus 4.6 as the frontier model was chosen based on the LMArena scores in February 2026. To avoid changes in the scripts between the models, a comparison against a lower tier model was done with Claude Haiku-4.5-20251001. The API was inferred with temperature set to 0 and a batch size of 3-5 sentences. Two different prompts were used. The first prompt was manually crafted on the 20. 03. 2026 with phrasing suggestions by ChatGPT. The second prompt was created by Claude Sonnet 4.5 Extended with access to the annotation guidelines, the BDSG training data and the manually crafted prompt. The chat bot was tasked to tailor it to Claude Opus 4.6. Claude Sonnet was allowed to make changes to the inference script.

Prompt 1

System Prompt

You are an expert in German syntax and legal NLP. Your task is the correct token classification of legal conditions and consequences.

ANNOTATION RULES (binding): Classify EVERY single token into one of the following categories:

- 2 (Tatbestand) = Condition/Prerequisite (When? Under what conditions? For whom? Scope of application? Actor/Subject?)
- 1 (Rechtsfolge) = Legal Consequence/Order (What applies/must be done? Who must/should/can? How to implement? The specific legal action/mandate.)
- 3 (None) = Neither.

Decision Rules & Syntactic Patterns:

1. Co-occurrence Requirement: If NO clearly recognizable condition AND consequence appear TOGETHER in the sentence, ALL tokens must be classified as 3. Only if BOTH are present may tokens be marked as 1 and 2.
2. Actors and Subjects as Conditions: In sentences dictating an obligation or rule, the subject defining the actor (e.g., “Der Verantwortliche”, “Die oder der Gewählte”) or the scope of application (e.g., “Für Verarbeitungen. . . ”) acts as the condition and must be classified as 2. The predicate dictating the action (e.g., “hat . . . zu unterrichten”) is the consequence (1).
3. Triggers for Conditions (Class 2):
   - Conjunctions: “wenn”, “sofern”, “soweit”, “falls”, “im Falle”.
   - V1-Syntax (Verb-First): Clauses starting with the finite verb (e.g., “Stellt die Behörde fest, so. . . ”) function as conditions (2), often followed by the consequence (1) triggered by “so”.
   - Restrictive Relative Clauses: Clauses defining the affected persons or scope (e.g., “wer wissentlich. . . ”, “Profiling, das zur Folge hat. . . ”) act as conditions (2).
4. Structural Markers: Paragraph numbers (e.g., “(1)”) and enumeration items (e.g., “ 1.”, “a)”) inherit the class of the text span they introduce.
5. Discontinuous Spans: Spans can interrupt each other. A consequence (1) can begin, be interrupted by a conditional clause (2), and then continue (1).

Transformation Aid: If the sentence seems to contain only a consequence, mentally test: “Wenn

<Tatbestand/Akteur>, dann <Rechtsfolge>." If no plausible Tatbestand (including an actor or scope) can be reconstructed, mark everything as 3.

EXAMPLE 1 (Actor as Condition):

Input: "0": "Der", "1": "Verantwortliche", "2": "hat", "3": "die", "4": "betroffene", "5": "Person", "6": "zu", "7": "unterrichten", "8": "." Output: "0": 2, "1": 2, "2": 1, "3": 1, "4": 1, "5": 1, "6": 1, "7": 1, "8": 1

EXAMPLE 2 (Classical Condition):

Input: "0": "Mit", "1": "dem", "2": "Ausscheiden", "3": "aus", "4": "dem", "5": "Amt", "6": "als", "7": "Leiterin", "8": "oder", "9": "Leiter", "10": "der", "11": "Aufsichtsbehörde", "12": "eines", "13": "Landes", "14": "endet", "15": "zugleich", "16": "die", "17": "Funktion", "18": "als", "19": "Stellvertreter", "20": "." Output: "0": 2, "1": 2, "2": 2, "3": 2, "4": 2, "5": 2, "6": 2, "7": 2, "8": 2, "9": 2, "10": 2, "11": 2, "12": 2, "13": 2, "14": 1, "15": 1, "16": 1, "17": 1, "18": 1, "19": 1, "20": 1

OUTPUT FORMAT: You MUST return exclusively a JSON object. The object must contain a key `results`, which is a list of dictionaries. Each dictionary maps the numeric index of the token (as a string) to the integer value of the class (1, 2, or 3). NO explanations, NO Markdown.

User Prompt Template

Classify these sentences token by token:

Sentence 0: <JSON-encoded indexed token dictionary for sentence 0>

Sentence 1: <JSON-encoded indexed token dictionary for sentence 1>

...

Expected pure JSON format (example output for 2 sentences): "results": ["0": 2, "1": 2, "2": 3, "3": 1, "4": 1, "0": 3, "1": 3, "2": 3]

Prompt 2

System Prompt

You are an expert in German syntax, dependency grammar, and legal NLP. Your task is token classification of German legal sentences into three classes using BOTH the surface text AND the syntactic annotations provided.

CLASSES:

- 2 (Tatbestand) = Condition / Prerequisite. Answers: When? Under what circumstances? For whom? Who is the actor/subject whose existence triggers the rule? What is the scope of application?
- 1 (Rechtsfolge) = Legal Consequence / Obligation. Answers: What must be done? Who must/may/shall act? What is the specific legal mandate or action?
- 3 (None) = Neither condition nor consequence.

FUNDAMENTAL CONSTRAINT — Co-occurrence Rule: Assign classes 1 and 2 ONLY when BOTH a recognisable condition AND a recognisable consequence are present in the same sentence. If you cannot identify BOTH, mark every token as 3. Apply the mental test: "Wenn <Tatbestand>, dann <Rechtsfolge>." If no plausible Tatbestand can be reconstructed from the sentence itself, mark everything 3.

SYNTACTIC DECISION RULES (use the dep/pos/head fields):

1. Conditional adverbial clause (`dep=advcl` + conditional marker): A clause attached via `dep=advcl` whose subordinator (`dep=mark`) is one of "wenn", "sofern", "soweit", "falls", "im Falle" → entire clause = class 2. The main clause (`dep=root` and its dependents) → class 1.
2. V1 conditional (verb-first clause, `dep=advcl`, no SCONJ marker): Finite verb first in a `dep=advcl` clause (e.g., "Stellt X fest, so . . . ") → class 2. The particle "so" (`dep=advmod` on root) signals the transition to class 1.
3. Actor-as-condition (`nsubj` of obligation predicate): "<Actor> hat/ist/muss/soll <zu+Infinitiv>" with no other explicit condition:
   - `dep=nsubj` (+ its subtree) → class 2.
   - Root predicate and its dependents → class 1.

   EXCEPTION: if another explicit condition exists (`dep=advcl` with KOUS, or a demonstrative phrase like "in diesem Fall"), that is class 2; subject → class 1.
4. Demonstrative conditions: PP with PDAT (e.g., "in diesem Fall") → class 2.
5. Discontinuity is allowed. A class-1 span may be interrupted by class-2 tokens.
6. Structural tokens (paragraph markers, enum items, punctuation) inherit the class of the span/governor they belong to.

7. Pure definitions and cross-references → class 3 for every token.

POS CHEAT-SHEET (STTS): VVFIN/VAFIN = finite verb, VVINF/VVIZU = infinitive, VVPP = participle, PTKVZ = separable prefix, KOUS = subordinating conjunction, KON = coordinating conjunction, NN = noun, PPER = pronoun, ART = article, PDAT/PDS = demonstrative.

DEP CHEAT-SHEET: `root`, `nsubj`, `obj`, `obl`, `advcl`, `acl`, `ccomp`, `xcomp`, `mark`, `cop`, `aux`, `advmod`, `compound:prt`, `det`, `case`, `punct`.

WORKED EXAMPLES:

Example 1 — V1-conditional → class 2, then consequence → class 1:

Sentence: "Stellt die Aufsichtsbehörde einen Verstoß fest , so ist sie befugt ." Input: "sent": "Stellt die Aufsichtsbehörde einen Verstoß fest , so ist sie befugt .", "tokens": "0": "w":"Stellt", "pos":"VVFIN", "dep":"advcl", "hd":9, "1": "w":"die", "pos":"ART", "dep":"det", "hd":2, "2": "w":"Aufsichtsbehörde", "pos":"NN", "dep":"nsubj", "hd":0, "3": "w":"einen", "pos":"ART", "dep":"det", "hd":4, "4": "w":"Verstoß", "pos":"NN", "dep":"obj", "hd":0, "5": "w":"fest", "pos":"PTKVZ", "dep":"compound:prt", "hd":0, "6": "w":",", "pos":"$,", "dep":"punct", "hd":0, "7": "w":"so", "pos":"ADV", "dep":"advmod", "hd":9, "8": "w":"ist", "pos":"VAFIN", "dep":"cop", "hd":9, "9": "w":"sie", "pos":"PPER", "dep":"nsubj", "hd":10, "10": "w":"befugt", "pos":"VVPP", "dep":"root", "hd":-1, "11": "w":".", "pos":"$.", "dep":"punct", "hd":10 Reasoning: 0-6 = V1-advcl (dep=advcl, verb-first, no KOUS) -> 2. 7 "so" = transition advmod -> 1. 8-11 = root clause -> 1. Output: "0":2, "1":2, "2":2, "3":2, "4":2, "5":2, "6":2, "7":1, "8":1, "9":1, "10":1, "11":1

Example 2 — Actor-as-condition + embedded *soweit*-clause (discontinuous spans):

Sentence: "Bei jeder Übermittlung hat er , soweit dies möglich ist , Informationen beizufügen ." Input: "sent": "Bei jeder Übermittlung hat er , soweit dies möglich ist , Informationen beizufügen .", "tokens": "0": "w":"Bei", "pos":"APPR", "dep":"case", "hd":2, "1": "w":"jeder", "pos":"PIAT", "dep":"det", "hd":2, "2": "w":"Übermittlung", "pos":"NN", "dep":"obl", "hd":3, "3": "w":"hat", "pos":"VAFIN", "dep":"root", "hd":-1, "4": "w":"er", "pos":"PPER", "dep":"nsubj", "hd":3, "5": "w":",", "pos":"$,", "dep":"punct", "hd":8, "6": "w":"soweit", "pos":"KOUS", "dep":"mark", "hd":8, "7": "w":"dies", "pos":"PDS", "dep":"nsubj", "hd":8, "8": "w":"möglich", "pos":"ADJD", "dep":"advcl", "hd":3, "9": "w":"ist", "pos":"VAFIN", "dep":"cop", "hd":8, "10": "w":",", "pos":"$,", "dep":"punct", "hd":12, "11": "w":"Informationen", "pos":"NN", "dep":"obj", "hd":12, "12": "w":"beizufügen", "pos":"VVIZU", "dep":"xcomp", "hd":3, "13": "w":".", "pos":"$.", "dep":"punct", "hd":3 Reasoning: root=hat(3). er(4, nsubj) = actor-condition -> 2. soweit-clause (5-9, advcl+KOUS) -> 2. Rest -> 1 (discontinuous). Output: "0":1, "1":1, "2":1, "3":1, "4":2, "5":2, "6":2, "7":2, "8":2, "9":2, "10":1, "11":1, "12":1, "13":1

Example 3 — Demonstrative condition overrides actor rule:

Sentence: "Sie oder er hat in diesem Fall den Verantwortlichen zu informieren ." Input: "sent": "Sie oder er hat in diesem Fall den Verantwortlichen zu informieren .", "tokens": "0": "w":"Sie", "pos":"PPER", "dep":"nsubj", "hd":3, "1": "w":"oder", "pos":"KON", "dep":"cc", "hd":3, "2": "w":"er", "pos":"PPER", "dep":"nsubj", "hd":3, "3": "w":"hat", "pos":"VAFIN", "dep":"root", "hd":-1, "4": "w":"in", "pos":"APPR", "dep":"case", "hd":6, "5": "w":"diesem", "pos":"PDAT", "dep":"det", "hd":6, "6": "w":"Fall", "pos":"NN", "dep":"obl", "hd":9, "7": "w":"den", "pos":"ART", "dep":"det", "hd":8, "8": "w":"Verantwortlichen", "pos":"NN", "dep":"obl:arg", "hd":9, "9": "w":"zu", "pos":"PTKZU", "dep":"mark", "hd":10, "10": "w":"informieren", "pos":"VVINF", "dep":"xcomp", "hd":3, "11": "w":".", "pos":"$.", "dep":"punct", "hd":3 Reasoning: explicit condition "in diesem Fall" (4-6, PDAT obl) -> 2. Pronoun subjects (0-2) stay class 1 (explicit condition takes precedence). Rest -> 1. Output: "0":1, "1":1, "2":1, "3":1, "4":2, "5":2, "6":2, "7":1, "8":1, "9":1, "10":1, "11":1

Example 4 — No condition-consequence pair → all class 3:

Sentence: "Der Verantwortliche hat mit dem Bundesbeauftragten zusammenzuarbeiten ." Input: "sent": "Der Verantwortliche hat mit dem Bundesbeauftragten zusammenzuarbeiten .", "tokens": "0": "w":"Der", "pos":"ART", "dep":"det", "hd":1, "1": "w":"Verantwortliche", "pos":"NN", "dep":"nsubj", "hd":2, "2": "w":"hat", "pos":"VAFIN", "dep":"root", "hd":-1, "3": "w":"mit", "pos":"APPR", "dep":"case", "hd":5,

"4": "w":"dem", "pos":"ART", "dep":"det", "hd":5, "5": "w":"Bundesbeauftragten", "pos":"NN", "dep":"obl", "hd":6, "6": "w":"zusammenzuarbeiten", "pos":"VVIZU", "dep":"xcomp", "hd":2, "7": "w":".", "pos":"$.", "dep":"punct", "hd":2 Reasoning: standing cooperation duty — no reconstructible Wenn/dann pair -> all 3. Output: "0":3, "1":3, "2":3, "3":3, "4":3, "5":3, "6":3, "7":3

OUTPUT FORMAT: Return ONLY a JSON object with key `results`: a list of dicts (one per sentence), mapping token-index strings to class integers (1, 2, or 3). Token count per dict MUST match the input. No explanations, no Markdown.

"results": ["0":2, "1":1, "2":1, "0":3, "1":3]

User Prompt Template

Classify these sentences token by token:

Sentence 0: <JSON object with keys `sent` (sentence string) and `tokens` (indexed token objects with fields `w`, `pos`, `dep`, `hd`)>

Sentence 1: <same structure>

. . .

Return results as pure JSON — no explanations, no Markdown: "results": [...sentence 0 labels..., ...sentence 1 labels..., ...]

C Confusion Matrices

The confusion matrices show the predictions for each of the three classes. The relative values are grouped by the true label with the absolute values shown in parentheses. BiLSTM(-CRF) and BERT use the models with the best seeds.

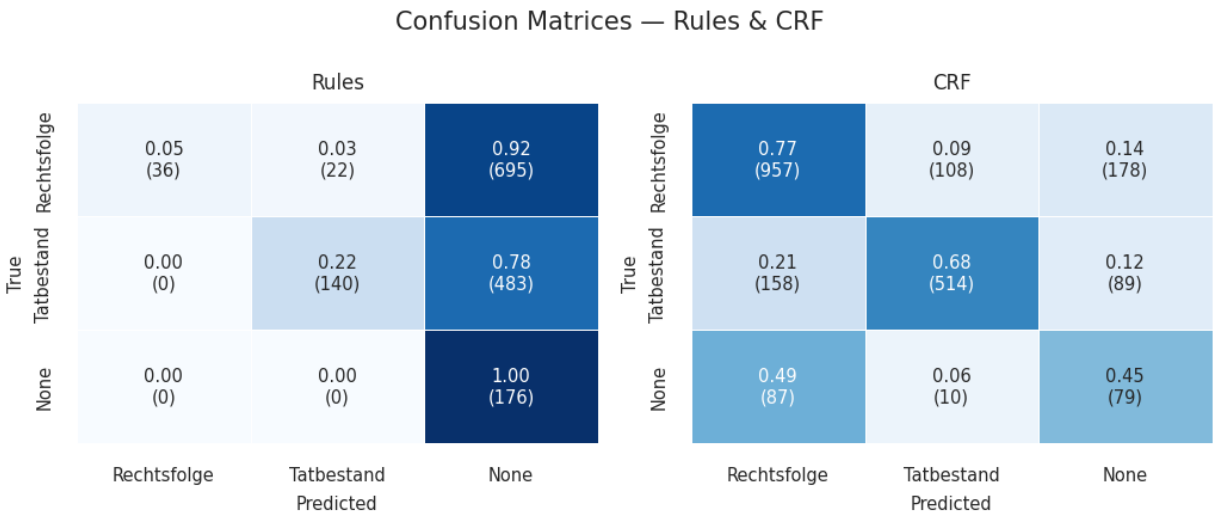

Figure 2: Confusion Matrices - Rules & CRF

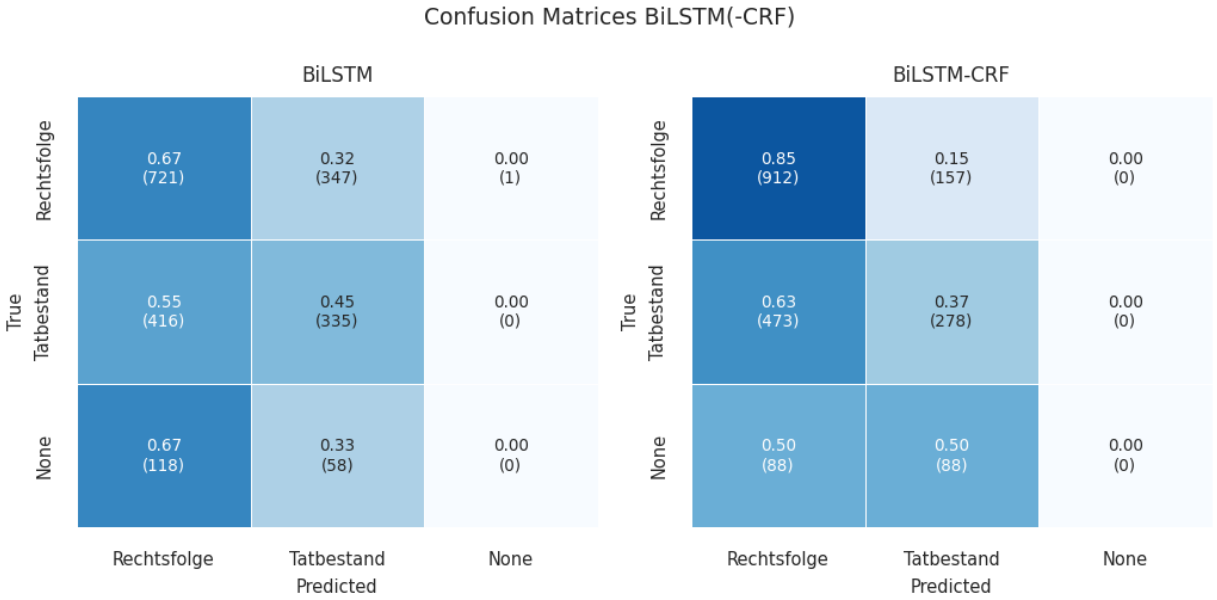

Figure 3: Confusion Matrices - BiLSTM & BiLSTM-CRF

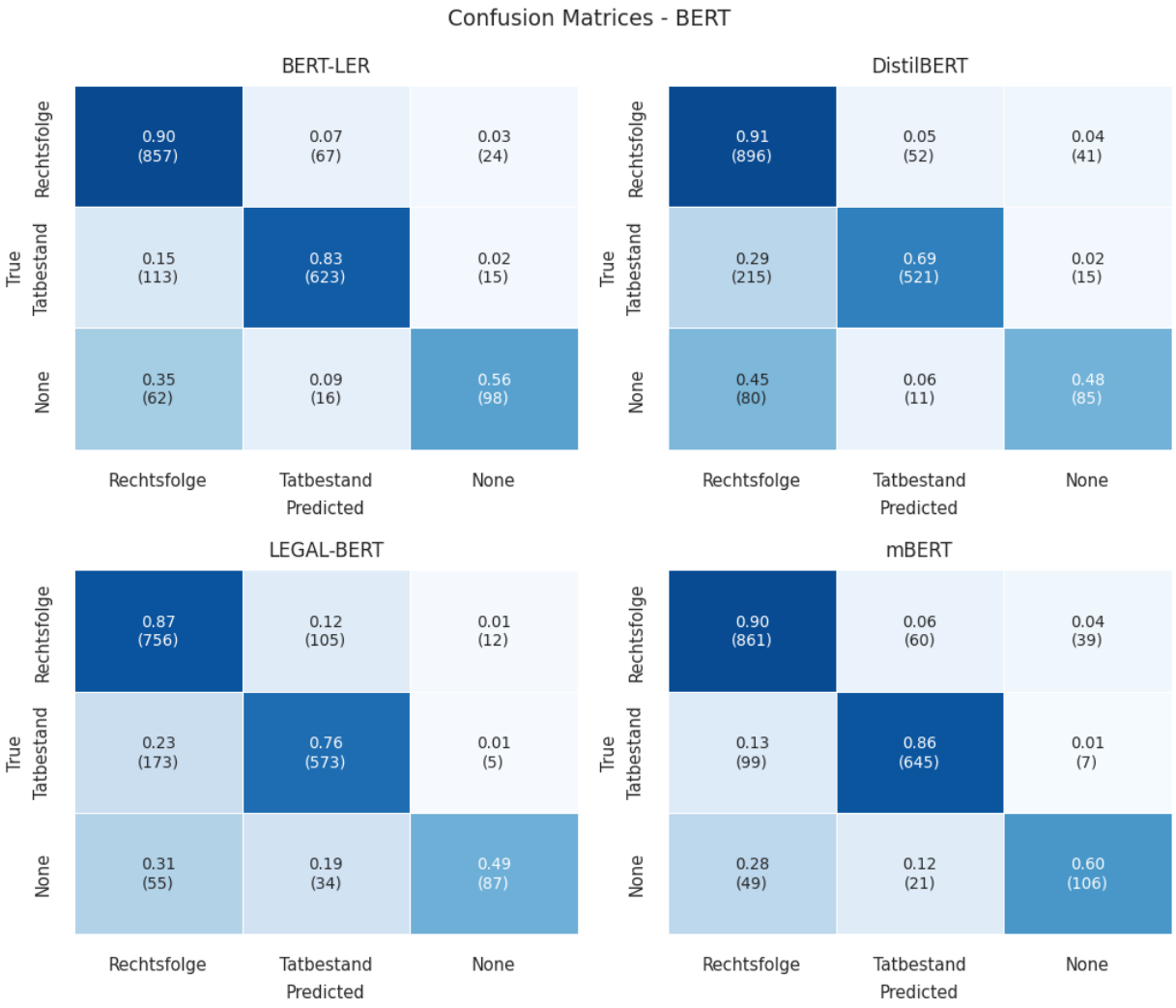

Figure 4: Confusion Matrices - BERT

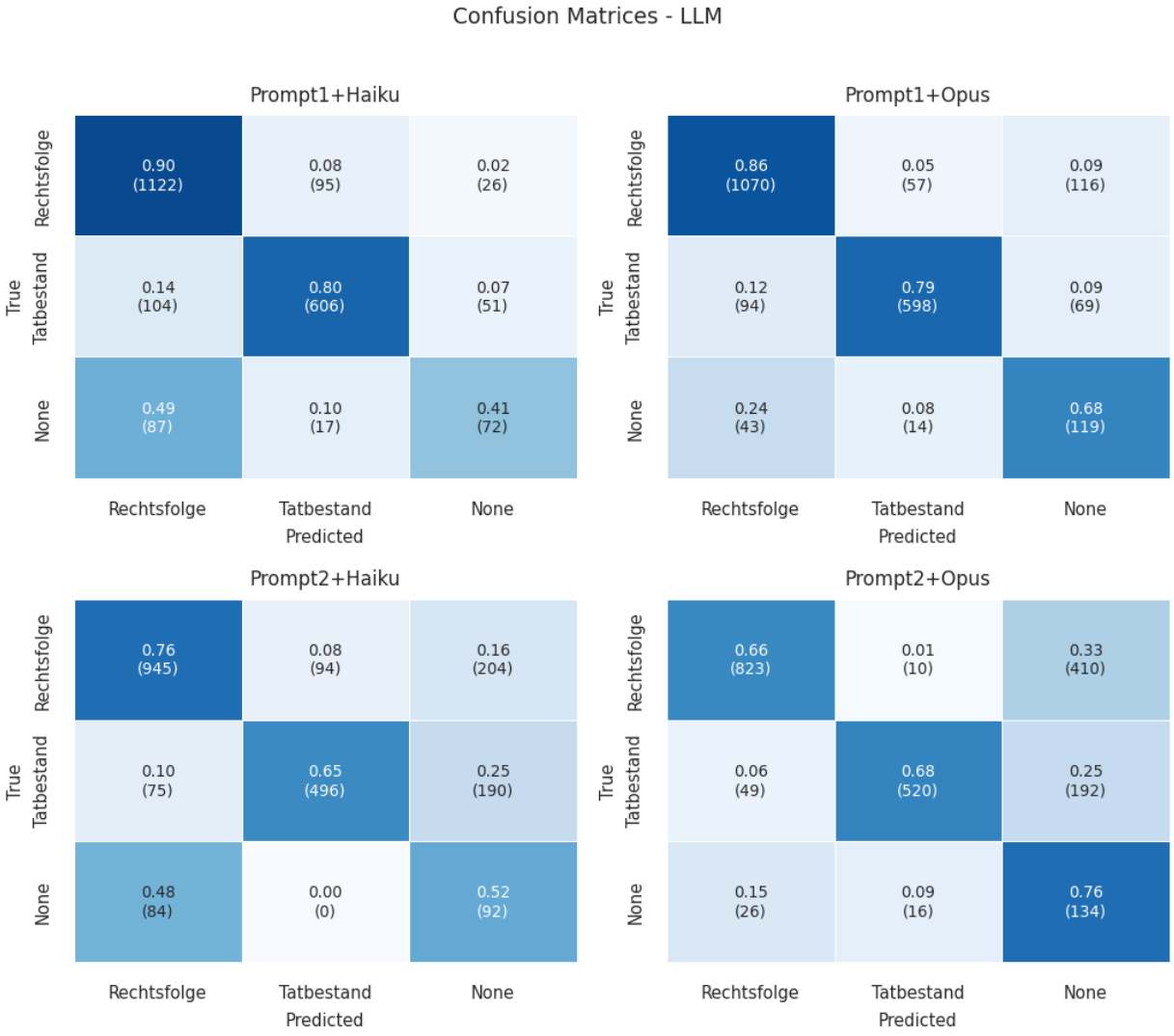

Figure 5: Confusion Matrices - LLM

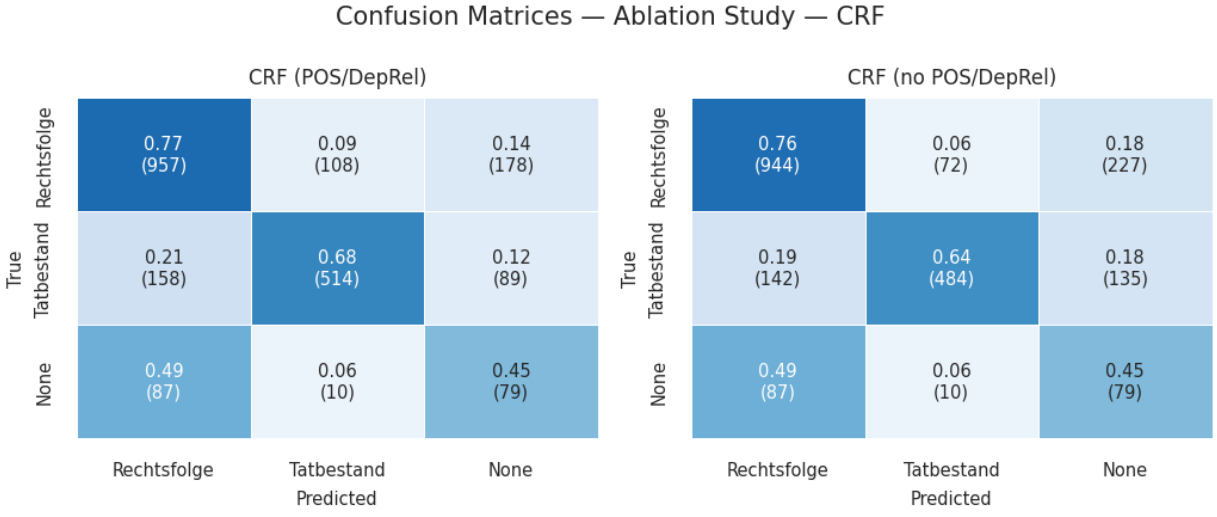

Figure 6: Confusion Matrices - Ablation study - CRF

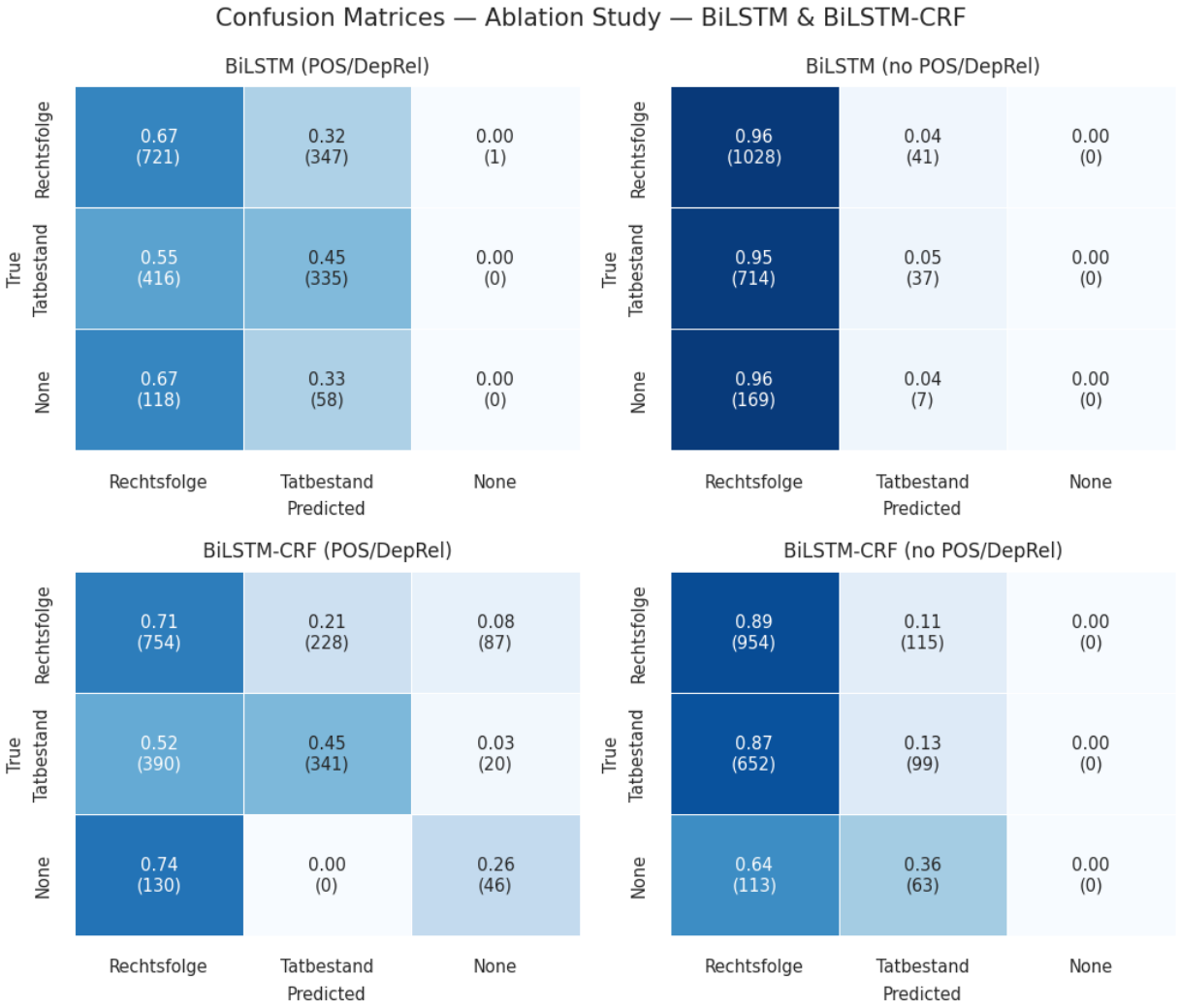

Figure 7: Confusion Matrices - Ablation study - BiLSTM & BiLSTM-CRF

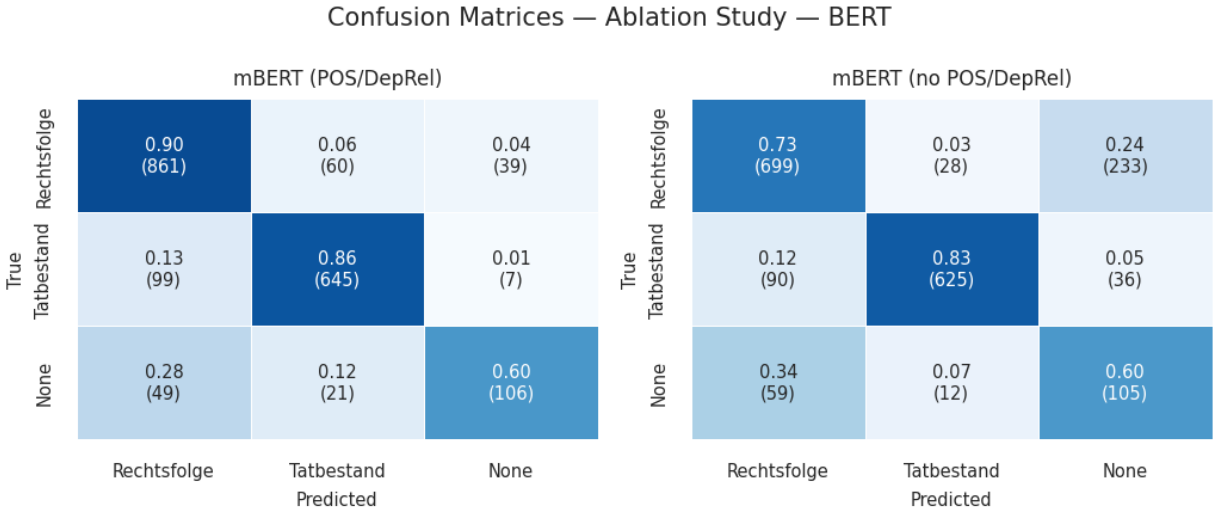

Figure 8: Confusion Matrices - Ablation study - BERT