Verifiable Retrieval for Legal Reasoning through Structure-driven Knowledge Graph

Ibrahim Siddig[1,*], Sviatoslav Tugeev[1] and Munir Georges[1]

[1]Technische Hochschule Ingolstadt, Esplanade 10, 85049 Ingolstadt, Germany

Abstract

Any automation within reliability-critical contexts such as medicine or law has to be transparent and verifiable. Despite their advantages, current LLM-based systems do not meet these requirements due to their inherent black-box nature. Retrieval Augmented Generation (RAG) attempts to ground the generative process in the relevant interpretable documents, thereby reducing the risk of AI 'hallucinations' and potentially enabling verification of the initial generation context. However, as the number and size of potential sources increase, naive retrieval approaches tend to perform worse, ignoring critical context elements. In this paper we aim to tackle this problem in the context of German Legal Code. Using structure-driven parsing to create a Knowledge Graph (KG) for the source document navigation and retrieval, we compare several KG-informed chunking and retrieval strategies and evaluate their effect on the accuracy of retrieved citations.

Keywords

Knowledge graph, Legal reasoning, Structure-driven parsing, Transparency in AI, AI in legal systems, Graph-RAG architecture

1. Introduction

The practical quality of legal answers is determined not only by their linguistic coherence and relevance to the context of the question, but by the degree to which they are supported by the relevant legal documents. This support is usually demonstrated by citing specific provisions, enabling the end user to verify legal validity of the answer. In practice, a legal text that is linguistically correct but cites the wrong sources or cites them incorrectly is not just useless, but harmful, as it looks trustworthy while being factually wrong or incomplete. This is especially relevant in the context of AI systems, which are already known for their ability to produce false but extremely plausible results [1]. In the context of RAG, where the output is generated based on the relevant law texts, this problem can be separated into three key issues:

- Whether all source documents or their segments which contain information necessary for a valid legal judgment are retrieved based on the initial query (factual description of the case).
- Whether the generated answer corresponds to the facts of the case and the content of the retrieved documents.
- Whether the retrieved documents are cited correctly within the final answer.

For German law, key prerequisites for studying this topic have been available for some time. The OpenLegalData corpus [1] [2, 3] provides 424 000 German court decisions with structured metadata, separated into the following sections: **Tenor** which explains the final decision, **Tatbestand** which shows statements of case facts; and **Entscheidungsgründe** which states the reasoning behind the final case decision. This extensive data is, however, insufficient, without a reliable way of processing it. This paper seeks to address the first issue in the RAG pipeline by exploring the viability of a structure-driven knowledge graph for addressing accuracy and token efficiency of retrieval in legal AI. In order to do that, we attempt to answer the following research questions:

RAGE-KG 2026: Retrieval-Augmented Generation Enabled by Knowledge Graphs, October 25–26, 2026, Bari, Italy

*Corresponding author.

Ibrahim.Siddig@thi.de (I. Siddig); Sviatoslav.Tugeev@thi.de (S. Tugeev); Munir.Georges@thi.de (M. Georges)

© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

[1]https://huggingface.co/datasets/harshildarji/openlegaldata

1. **RQ1:** How does a structure-driven KG influence the retrieval relevance and citation precision in the RAG pipeline for legal documents?
2. **RQ2:** What level of granularity in structure-aware text chunking is most beneficial for the relevance of retrieved sources and the overall retrieval efficiency?
3. **RQ3:** How do different one-hop graph-driven chunk expansion strategies affect retrieval accuracy across different chunking levels?

1.1. Key contributions

1. **A knowledge graph of German federal statute law.** 58 codes, an eight-level typed hierarchy from root down to *Unterbuchstabe* which represents the atomic level of the German federal law, 168 000 nodes, and 240 000 edges of which 22 000 are intra-code cross-references derived from in-statute citation tagging. The knowledge graph covers 90.3% of the ground truth citations in the OpenLegalData BGH evaluation set. The remaining 9.7% are out-of-scope or unretrievable.
2. **A structured token-normalized citation-faithfulness metric.** We evaluate retrieved citations using full hierarchical path matching (`Buch / § / Abs. / Satz / Nr. / Buchst.`)[2]. Leveraging the KG's hierarchy Fig. 1, the metric evaluates accuracy depth-wise to distinguish granular errors, such as citing an incorrect sentence within a paragraph versus citing an entirely wrong legal code. To decouple evaluation from chunk size bias, we normalize recall against total retrieved context length (1).
3. **Detailed comparison of chunking and one-hop retrieval strategies.** On a held-out set of 12 037 BGH cases, we compare different levels of granularity in law text chunking and one-hop KG-based retrieval strategies using the proposed metrics.

2. Related Work

[2] released 250 000 German court decisions with structured metadata; the subsequent segmentation work of [3] released a cleaned JSONL corpus of 251 038 decisions with Tenor / Tatbestand / Entscheidungsgründe splits. The *Bundesrecht* library [4] covers the upstream task of parsing German statutory reference strings, providing a structured JSONL corpus of 6873 federal laws. Our knowledge graph is complementary rather than competing. While *Bundesrecht* optimizes for citation-string parsing and depth-aware lookup over a flat representation, we build a graph with defined cross-reference edges.

[5] proposed a hierarchical and temporal graph-RAG architecture for legal norms but did not evaluate on real court decisions. [6] presented NyayaRAG, a statute-plus-case RAG framework for Indian legal judgment prediction; their evaluation conflates flat statute retrieval and case retrieval. [7] at RAGE-KG 2025 benchmarked KG-based RAG systems on legal documents. Our work differs in scoping the retrieval knowledge base to statute law alone, holding case law out entirely as evaluation gold, and measuring citation faithfulness at the level courts actually cite.

Prior evaluations of legal citation accuracy typically match at a flat or abstract level [8, 9], which cannot distinguish § 574 Abs. 2 Nr. 2 from § 574 Abs. 3. Our structured metric closes this gap by accounting for strict hierarchical subdivisions critical for legal documents. Empirical evaluations demonstrate that language models fail to robustly utilize long inputs, exhibiting severe performance degradation when critical information is positioned in the middle of the context window [10]. Modern frameworks increasingly employ multi-level hierarchical segmentation, ensuring semantic continuity across diverse text scales while preserving structural layout [11]. The necessity of token-efficient graph retrieval is further highlighted by TERAG [12], which adapts Personalized PageRank [13] to cut GraphRAG token costs down to 3%–11% while maintaining retrieval fidelity. In a similar spirit of scaling graph-based retrieval to large legal corpora, our work demonstrates that optimizing node granularity directly tackles this token bottleneck, delivering high recall efficiency per token.

[2] `Buch`: Book/Code; §: Section (*Paragraph*); `Abs.`: Subsection (*Absatz*); `Satz`: Sentence; *`Nr.`*: Number (*Nummer*); `Buchst.`: Letter (*Buchstabe*).

3. Methodology

3.1. Knowledge graph construction

To build our KG, a set of 58 codes was compiled based on the citations in the *Entscheidungsgründe* and law references sections of the OpenLegalData corpus. The full hierarchy, from the code down to the *Buchstabe*, is extracted from the official consolidated text published at *gesetze-im-internet.de* using our custom regex parser, established in [14] and expanded here with additional patterns to enable processing of documents with more diverse structures. Citations occurring within a statute are tagged during parsing and represented as directed `cites_*` edges; all 22,253 of them resolve within a single code, so the result is a forest of 58 disjoint code graphs. The resulting KG in Fig. 1 includes the following hierarchical node types, representing the structure of the original document: ***Buch*** · ***Titel*** · ***§*** · ***Absatz*** · ***Satz*** · ***Nummer*** · ***Buchstabe*** [3]

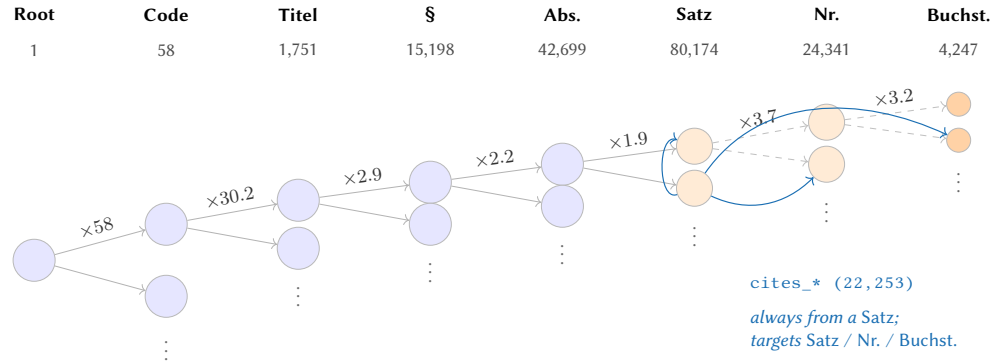


Figure 1: The Statute knowledge graph hierarchy includes 58 federal codes, 168,469 nodes, and 240,507 edges. Column headers display each level's name and total node count, noting that not all *Sätze* contain a *Nummer* or *Buchstabe*. Edge labels provide the average number of child nodes per branching parent at that level. Cross-reference edges marked `cites_*`, drawn in blue. Citations originate at a *Satz* node since they are tagged in sentence text and target a *Satz* (15,030), *Nummer* (6,805) or *Buchstabe* (418) node within the same code. The final graph is a forest of 58 disjoint code graphs, connected to the empty parent root node.

Each node is linked to the associated segment of text. The parent node contains the text of all its child nodes, i.e. if an *Absatz* node has multiple *Satz* child nodes, text associated with the *Absatz* node includes text of all the child *Satz* nodes. The constructed KG consists of the single root node not related to any specific document, which then diverges into multiple subtrees, each representing specific legal code. While there are no connecting edges between the subtrees, the decision to unify them into a single graph was guided by the fact that in the OpenLegalData corpus multiple codes are regularly cited together, demonstrating that legal decisions require access to an arbitrary number of codes simultaneously. Unlike the framework described in [5], the resulting KG includes only two types of edges: `contains_*`, to represent the hierarchical structure of the document, and `cites_*`, to represent citation references. The temporal dimension, usually expressed via `supersedes_*` or `amends_*` is not relevant to the selected documents.

3.2. Experiment setup

All text segments, including statutory chunks and query fields, are mapped to a continuous vector space using `KaLM-Embedding-Gemma3-12B-2511` [15]. This 12-billion parameter text embedding and retrieval model is optimized for high-density retrieval and achieves state-of-the-art multilingual performance on benchmarks such as MMTEB.

To evaluate the impact of different chunking granularities and multi-hop graph retrieval strategies, we establish the pipeline illustrated in Fig. 2.

[3] ***Buch***: Book/Code; ***Titel***: Title; ***§***: Section (*Paragraph*); ***Absatz***: Subsection; ***Satz***: Sentence; ***Nummer***: Number; ***Buchstabe***: Letter.

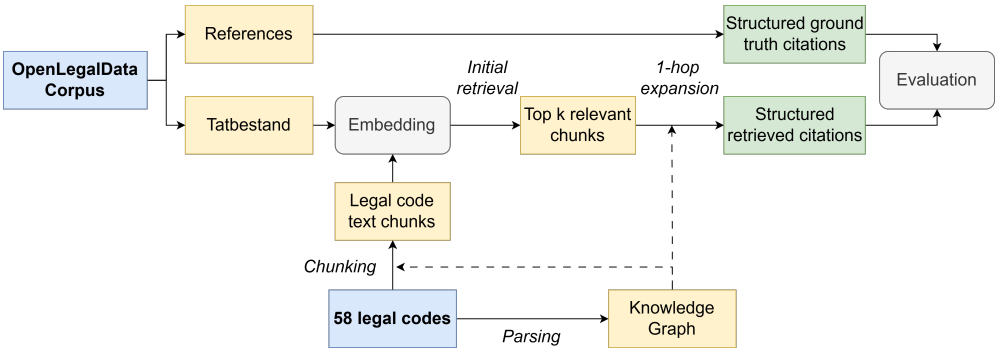


Figure 2: After the initial chunking, whether naive or KG-driven, the resulting chunks are embedded using KaLM-Embedding-Gemma3-12B. For each entry in the OpenLegalData corpus, the *Tatbestand* section, which describes the facts of the case, is embedded the same way to serve as pipeline input, and the top-k relevant chunks of the legal codes are retrieved. This list is then optionally expanded through a 1-hop expansion strategy using KG. The final structured list of retrieved references is compiled based on the IDs of associated nodes in KG and compared to the structured list of references from the corresponding section of the same corpus entry.

3.2.1. Chunking strategies

Structure-driven KG enables granular chunking of the original document, selecting chunks based on different node levels. Different chunking sizes offer a tradeoff between the probability that any retrieved chunk will include relevant information and the increasing context window, coupled with a reduced effect of any single chunk on the final model output. To compare different chunking strategies for processing legal documents using KG, in this paper we explored the effectiveness of chunking on the following levels, listed in increasing level of granularity: **§** · **Absatz** · **Satz** · **Nummer** · **Buchstabe**[4]

These levels were compared to each other and against a naive chunking approach, which did not utilize KG and simply chunked the texts of the legal codes into consecutive 100-word chunks, ignoring the document structure. The constant 100-w size was chosen to ensure structure-agnostic approach.

3.2.2. One-hop retrieval strategies

As each text chunk is linked to a specific node, a set of relevant text chunks can be further expanded using KG edges. As KG includes two types of edges, we evaluated three different one-hop expansion strategies:

- **Parent node.** This strategy utilized only the `contains_*` edge type, representing hierarchical links in KG. As a result, this expansion included the texts of parent nodes of the initially retrieved top-k.
- **Citation.** This strategy utilized only the `cites_*` edge type, representing reference links in KG, including the nodes cited directly in the initially retrieved top-k.
- **Hybrid.** This strategy utilized both types of links at the same time.

These strategies were compared to the baseline where only the originally retrieved top-k nodes were used.

3.2.3. Citation Matching and Evaluation Metric

Since the paper is focused on a retrieval-only segment of RAG pipeline, we compare the set of retrieved citations contained in the top-k chunks (both initial and expanded) against the case's ground truth citations. Both sets are normalized into an ordered tuple over the German statutory hierarchy to ensure consistent comparison: *Buch*[5] · *§* · *Absatz* · *Satz* · *Nummer* · *Buchstabe*

[4] ***§***: Section (*Paragraph*); ***Absatz***: Subsection; ***Satz***: Sentence; ***Nummer***: Number; ***Buchstabe***: Letter.
[5] ***Buch***: Book/Code.

Ground truth citations are parsed from the case references, while retrieved citations are decoded directly from the KG IDs of the retrieved nodes. To evaluate granular citation correctness, we report depth-stratified evaluation using a *prefix match rule*, where agreement is credited as long as all levels explicitly specified in both the retrieved and ground truth citations match. Unspecified deeper levels such as a missing *Nummer* or *Buchstabe* are counted as implicit hits. This mirrors the naturally heterogeneous judicial citation practices. For detailed statistics refer to Table 1.

Because raw recall (R) is monotonic in chunk size, as bigger chunks trivially cover more citations at a massive token cost, we instead apply a custom metric, **token efficiency recall (ToER) per** $1{,}000$ **words of chunk length** as defined in (1). Standard F1 score is insufficient, since precision falls as recall rises and the two effects cancel each other, reducing the score sensitivity to the extent of added context. To isolate retrieval quality from coverage constraints in Sect. 1.1, the ground truth is restricted to citations whose codes exist in the constructed knowledge graph.

$$\mathrm{ToER} = \frac{10^3}{W_\ell} \cdot \underbrace{\frac{1}{|\mathcal{G}|} \sum_{C_g \in \mathcal{G}} \max_{C_r \in \mathcal{R}_k} \mathrm{Match}(C_r, C_g)}_{\text{prefix recall}} \tag{1}$$

Where W_ℓ is the mean length of a single chunk at the granularity ℓ averaged over the *entire set of chunks in the whole corpus* at that granularity. It is thus a property of the chunking scheme rather than of any individual retrieval, and is constant across queries, across k, and across expansion strategies. $\mathcal{R}_k$ is the set of citations decoded from the retrieved nodes, $\mathcal{G}$ denotes the set of ground truth citations for the case and $\mathrm{Match}(C_r, C_g) \in \{0, 1\}$ evaluates the prefix-rule agreement between a retrieved citation C_r and a ground truth citation C_g. Both citations are truncated at the evaluation depth being reported. Table 1 presents the depth-stratified ToER performance. Reported values are micro-averaged across $12{,}037$ cases.

4. Results & Discussion

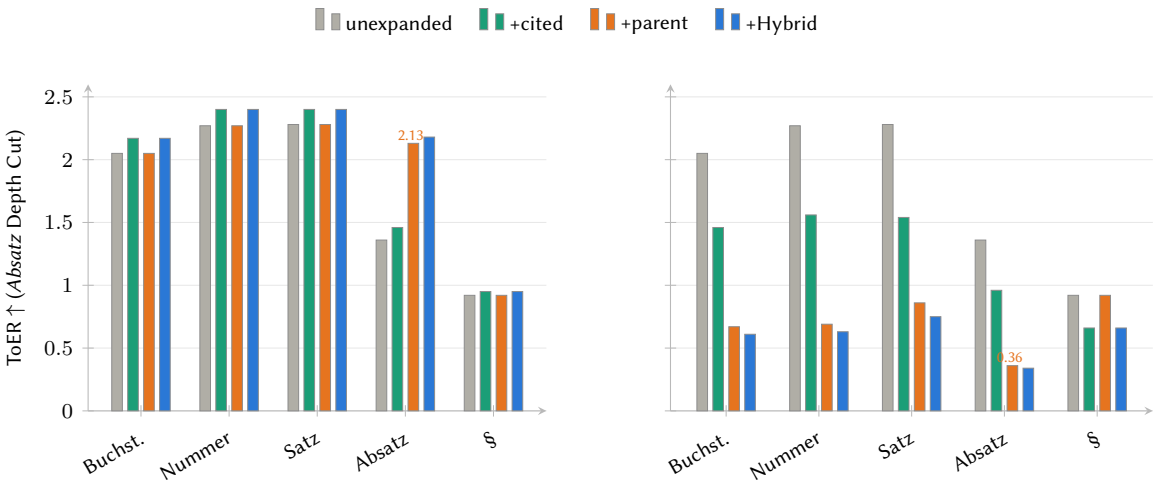
As shown in Table 1, structure-aware chunking strategy with higher granularity consistently dominates across all evaluation depths. Specifically, the **Nummer** chunking granularity achieves the highest ToER at the *Code* (14.23) and *§* (3.25) evaluation depths. Meaning that this approach retrieves the most relevant chunks of the source texts at the lowest token cost, when evaluated up to this depth. Chunking at **Satz** level yields the strongest performance at every deeper evaluation depth, reaching 2.28 at *Absatz*, 1.53 at *Satz*, and 2.79 at *Nummer*. Both configurations recover approximately three times more citations per 1,000 words than the **Naive-100w** chunking baseline, showing 14.23 vs. 4.19 at *Code*, 3.25 vs. 1.10 at *§*, and 2.79 vs. 0.81 at *Nummer* evaluation depths.

Notably, this performance increase is driven by the chunking granularity itself rather than merely introducing structural awareness. Chunking at higher levels, such as **§ (Paragraph)**, underperforms, even falling below the unstructured **Naive-100w** baseline with a ToER of only 3.06 at *Code* and 0.85 at *§* evaluation depths. Chunking at **Absatz** then represents an intermediate transition (7.78 and 1.93, respectively). Across all chunking granularities, ToER drops abruptly between the *Code* and *§* evaluation depth, such as **Nummer** falling from $14.23 \rightarrow 3.25$. This sharp drop exposes the heavy efficiency penalty the approach pays when moving from broad, code-level retrieval down to precise citations.

In Fig. 3, all evaluations are conducted at the *Absatz* evaluation depth, which is the depth at which ground truth citations most often terminate. It displays the effect of taking into account the token cost of context expansion. Figure 3a measures the ToER each strategy contributes without taking into account the token cost. The three expansion strategies discussed in Sect. 3.2.2 behave differently. The `cited` expansion strategy yields a small, uniform ToER gain across all chunking granularity levels, raising ToER $2.05 \rightarrow 2.17$ at **Buchstabe**, $2.27 \rightarrow 2.40$ at **Nummer**, and $2.28 \rightarrow 2.40$ at **Satz** chunking granularities. With the `parent` strategy, ToER score remains fixed across most chunking levels, except at **Absatz**, where the direct parent represents the entire statutory paragraph (§) and ToER increases

	Evaluation depth (strict match up to) [ToER ↑]					
Granularity (Chunks)	*Code*	*§*	*Absatz*	*Satz*	*Nummer*	*Buchst*
Buchstabe	12.89	2.93	2.05	1.35	1.91	0.50
Nummer	**14.23**	**3.25**	2.27	1.49	2.17	0.28
Satz	13.59	3.20	**2.28**	**1.53**	**2.79**	**0.52**
Absatz	7.78	1.93	1.36	1.37	1.57	0.30
§ (**Paragraph**)	3.06	0.85	0.92	0.90	0.94	0.16
Naive-100w (baseline)	4.19	1.10	0.85	0.72	0.81	0.30

Table 1
Depth-stratified ToER at top $k = 5$ retrieved chunks. Structure-aware chunking granularities (**Nummer** and **Satz**) yield the highest scores across all depths, outperforming both the **Naive-100w baseline** and **paragraph-level (§)** chunking strategies. Values in bold indicate the best-performing chunking granularity per evaluation depth.



(a) uncharged: $W{=}W_\ell$ (b) charged: $W{=}W_\ell^m$

Figure 3: Impact of 1-hop context expansion on token efficiency (ToER) at top-5 retrieved chunks. Evaluation is conducted under matching at the *Absatz* evaluation depth across all chunking granularities for unexpanded retrieval and three expansion strategies. Granularities are ordered from higher to lower, and the y-axis is shared between figures. Figure 3a is *uncharged* ($W{=}W_\ell$): expanded context is added to retrieval but not considered against the token budget. Figure 3b is *charged* ($W{=}W_\ell^m$): the expansion tokens count toward W, exposing the efficiency cost of each strategy.

from 1.36 to 2.13. At all chunking granularity levels except **Absatz**, `parent` expansion reproduces the baseline values exactly (2.05, 2.27, 2.28, and 0.92, respectively). Combining both expansion approaches in the `Hybrid` expansion further lifts the ToER score of the **Absatz** chunking granularity to 2.18.

Subsequently, context expansion cannot resolve a poor initial choice of chunking size. Even when we apply our most generous expansion (`Hybrid`) to the **Absatz** chunking depth, its performance (2.18) still falls short of smaller, `un-expanded` **Nummer** (2.27) or **Satz** (2.28) chunking depths. When a set of chunks retrieved after using these chunking strategies is expanded with the `Hybrid` approach, they both reach ToER $= 2.40$.

Figure 3b takes the token cost of context expansion into account. It replaces W_ℓ with W_ℓ^m, denoting the average total words delivered per chunk, which in this case means the base text plus appended hop context, averaged across the entire corpus. Every expansion strategy then falls below its own unexpanded baseline, and the largest apparent gain inverts: **Absatz** + `parent` moves the ToER $1.36 \rightarrow 0.36$, because adding § text raises the delivered context from 70.6 to 415.0 words. These findings reveal that the initial choice of chunking granularity remains the dominant factor in retrieval efficiency under either configuration.

5. Conclusion

This paper addressed the challenge of verifiable retrieval in legal AI by introducing a structure-driven Knowledge Graph (KG) approach for German statutory law. By evaluating various chunking and one-hop context expansion strategies, we demonstrated that aligning retrieval mechanisms with the inherent hierarchical structure of legal codes is essential for maximizing both citation recall and token efficiency.

Limitations Even under the best configuration, the absolute ToER is modest and decays steeply with evaluation depth. The strongest chunking granularities reach a ToER of ≈ 14.2 at the *Code* evaluation depth, but only ≈ 3.2 at the *§* and ≈ 1.5 at the *Satz* evaluation levels. This decay reflects the inherent difficulty of the task. This difficulty arises due to the domain gap between factual case descriptions (*Tatbestand*) and abstract legal norms. Consequently, dense retrieval embeddings struggle to bridge non-legal narrative language to precise statutory formulations without explicit domain adaptation.

Future Work The results of this paper can be expanded upon in multiple ways. First, a comparison between multiple embedders would provide valuable insight into the correlation between embedding strategy and the effectiveness of the KG-driven chunking approach. Second, structure-driven KG could benefit from the addition of entity-driven KG, which would encode additional relevance links within the document. Third, as different chunking levels utilize different proportions of context windows, additional experiments into comparative token efficiency with varied top-k retrieval between KG node levels are a valuable avenue of future research. Furthermore, while granular chunking improves retrieval efficiency, its direct effect on downstream generation quality and hallucination mitigation remains to be validated, given that this study explicitly isolates retrieval performance from downstream LLM artifacts. Lastly, evaluation of retrieved items by legal specialists with relevant domain knowledge would provide valuable insight into their sufficiency for informed legal judgment by a human specialist.

Data and code availability. The knowledge graph, the six chunk corpora, the evaluation set (BGH case identifiers with parsed ground-truth citations and a reconstruction script for the decision text), and all evaluation results are archived on Zenodo at https://doi.org/10.5281/zenodo.22110527. The reference implementation of the metric and the retrieval/evaluation pipeline is at https://github.com/ibrahimssd/structure-driven-legal-kg-rag.

Declaration on Generative AI

During the preparation of this work, the authors used Claude, Gemini, and Quillbot AI for the following purposes: grammar and spelling check, paraphrasing and rewording, and generating images through generating LaTeX code for Figure 1 to demonstrate non-generated data. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, Y. Ishii, Y. Bang, A. Marone, A. Madotto, P. Fung, Survey of hallucination in natural language generation, ACM Computing Surveys 55 (2023) 1–38.

[2] M. Ostendorff, T. Blume, S. Ostendorff, Towards an open platform for legal information, in: Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020, 2020, pp. 385–388.

[3] H. Darji, M. Heckelmann, C. Kratsch, G. de Melo, Segmentation and processing of german court decisions from open legal data, in: Legal Knowledge and Information Systems, IOS Press, 2025, pp. 276–281.

[4] H. Darji, M. Heckelmann, C. Kratsch, G. de Melo, Bundesrecht: An open library and corpus for german statutory reference processing, arXiv preprint arXiv:2605.31338 (2026).
[5] H. de Martim, Graph rag for legal norms: A hierarchical and temporal approach, arXiv e-prints (2025) arXiv–2505.
[6] S. K. Nigam, B. D. Patnaik, S. Mishra, A. V. Thomas, N. Shallum, K. Ghosh, A. Bhattacharya, Nyayarag: Realistic legal judgment prediction with rag under the indian common law system, in: Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, 2025, pp. 1709–1726.
[7] J. G. Ongris, F. Darari, B. C. Tobing, D. R. Faisal, O. Lee, Benchmarking kg-based rag systems: A case study of legal documents, in: CEUR Workshop Proceedings, volume 4079, CEUR-WS, 2025, pp. 69–82.
[8] S. Chen, H. Yin, S. Zhou, Legalcitebench: Evaluating citation reliability in legal language models, arXiv preprint arXiv:2605.10186 (2026).
[9] N. Guha, J. Nyarko, D. Ho, C. Ré, A. Chilton, A. Chohlas-Wood, A. Peters, B. Waldon, D. Rockmore, D. Zambrano, et al., Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models, Advances in neural information processing systems 36 (2023) 44123–44279.
[10] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, Lost in the middle: How language models use long contexts, Transactions of the association for computational linguistics 12 (2024) 157–173.
[11] W. Lu, K. Chen, Z. Shen, R. Qiao, X. Sun, Hichunk: Evaluating and enhancing retrieval augmented generation with hierarchical chunking, in: Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2026, pp. 29738–29753.
[12] Q. Xiao, H. T. Tsang, J. Bai, Terag: Token-efficient graph-based retrieval-augmented generation, arXiv preprint arXiv:2509.18667 (2025).
[13] B. J. Gutiérrez, Y. Shu, Y. Gu, M. Yasunaga, Y. Su, Hipporag: Neurobiologically inspired long-term memory for large language models, Advances in neural information processing systems 37 (2024) 59532–59569.
[14] I. Siddig, S. Tugeev, M. Georges, Pattern-based parsing of german traffic regulations (stvo) for legal knowledge graph construction, in: S. Grawunder (Ed.), Studientexte zur Sprachkommunikation: Elektronische Sprachsignalverarbeitung 2025, TUDpress, Dresden, 2025, pp. 255–264. URL: https://www.essv.de/pdf/2025_255_264.pdf.
[15] X. Zhao, X. Hu, Z. Shan, S. Huang, Y. Zhou, X. Zhang, Z. Sun, Z. Liu, D. Li, X. Wei, et al., Kalm-embedding-v2: Superior training techniques and data inspire a versatile embedding model, arXiv preprint arXiv:2506.20923 (2025).