JusBuild: a RAG-based Architecture for Legal Document Building - Discussion Paper

Silvana Castano[1,†], Alfio Ferrara[1,†], Stefano Montanelli[1,†], Sergio Picascia[1,*,†] and Davide Riva[2,†]

[1]*Università degli Studi di Milano, Department of Computer Science, Via Celoria, 18 - 20133 Milano, Italy*
[2]*Politecnico di Torino, Department of Control and Automation Engineering, Corso Duca degli Abruzzi, 24 - 10129 Torino, Italy*

Abstract

The creation of legal documents often requires the generation of text that adheres to a predefined schema, a process that can be significantly enhanced through the use of digital and automated tools. This paper presents JusBuild, a Retrieval-Augmented Generation (RAG)-based document builder architecture designed to assist legal practitioners in drafting new legal documents. JusBuild operates through a multi-layered framework: the Document Segmentation Layer partitions legal documents into functional sections based on a predefined schema; the Storage Layer collects semantically meaningful vector representations of these sections, generated by an embedding model; and the Retrieval and RAG Layers provide real-time suggestions to the practitioner during the drafting process. To evaluate its versatility, JusBuild was tested on two distinct datasets, varying in document template, language, and judicial matter, demonstrating its adaptability and applicability across diverse legal contexts. The results highlight JusBuild's potential to streamline legal document drafting while maintaining user full control over document production, leveraging its "human-in-the-loop" approach.

Keywords

Retrieval-Augmented Generation, Natural Language Processing, Legal Information Retrieval, Digital Justice

1. Introduction

Legal professionals deal with vast amounts of documentation, often navigating time-consuming and labor-intensive drafting processes. In recent years, Artificial Intelligence (AI) has emerged as a transformative force, enhancing accessibility and efficiency in legal workflows. Developing tools to support legal practitioners in drafting documents more effectively, however, is an unsolved challenge. Traditional methods require extensive research and precedent analysis, but Retrieval-Augmented Generation (RAG) offers a promising AI-driven solution. By leveraging Natural Language Processing (NLP), RAG can be exploited to retrieve relevant legal content and generate coherent, context-aware text, streamlining the drafting process.

In this paper, we introduce *JusBuild*, a framework architecture designed to provide assistance in legal document building, with a special interest in trial documents. JusBuild extends the *search-by-content* functionality presented in Castano et al. [1], performing a stream of tasks on input documents: document segmentation, storage, retrieval, and conditional text generation, powered by cutting-edge NLP models. *Document Segmentation* utilizes a Conditional Random Fields (CRF)-based model to identify functional sections within legal texts. *Storage* and *Retrieval* Layers employ an embedding model that captures semantic nuances, enabling effective indexing and search of legal text via a vector database.

JusBuild enhances legal drafting by offering AI-powered textual suggestions. It first *retrieves* relevant sections from past legal cases using semantic search and then refines these with AI-generated content via a Large Language Model (LLM) in a RAG pipeline. A key feature is the *human-in-the-loop* approach,

SEBD 2025: 33rd Symposium on Advanced Database Systems, June 16-19, 2025, Ischia, Italy
*Corresponding author.
†These authors contributed equally.
silvana.castano@unimi.it (S. Castano); alfio.ferrara@unimi.it (A. Ferrara); stefano.montanelli@unimi.it (S. Montanelli); sergio.picascia@unimi.it (S. Picascia); davide.riva@polito.it (D. Riva)
0000-0002-3826-2407 (S. Castano); 0000-0002-4991-4984 (A. Ferrara); 0000-0002-6594-6644 (S. Montanelli); 0000-0001-6863-0082 (S. Picascia); 0009-0003-9681-9423 (D. Riva)

ensuring that legal professionals maintain full control over the final output by selecting and refining AI-generated suggestions.

To validate JusBuild, we consider a dataset of first-instance civil law judgments from 12 courts of Northern Italy. Furthermore, we test the adaptability of JusBuild across jurisdictions using a second dataset with a different language, structure, and legal topic, i.e., a dataset of court decisions from the US Securities and Exchange Commission. JusBuild is currently a prototype tool under continuous development and refinement. An extended presentation of JusBuild as well as a discussion on possible applications of the proposed architecture are described in [2].

The paper is organized as follows: Related work is reviewed in Section 2. Sections 3 and 4 present the JusBuild framework architecture and related layers. Section 5 evaluates its effectiveness across different legal contexts, and Section 6 concludes with future research directions.

2. Related work

Our study integrates Retrieval-Augmented Generation (RAG) with legal document building, two evolving areas in legal technology.

Retrieval-Augmented Generation RAG enhances text generation by incorporating retrieved domain knowledge [3], compensating for LLMs' knowledge limitations. A typical RAG pipeline indexes domain data, retrieves relevant text chunks via semantic search, and augments user queries with retrieved information to prompt the LLM and generate responses.

RAG has been successfully applied across domains [4], including legal applications like legal question answering and document drafting. CBR-RAG [5] explores embedding models for legal question answering, while CaseGPT [6] employs case-based reasoning in legal and medical contexts. Addressing legal language complexity, ChatLaw [7] leverages a multi-agent model for professional legal consultations, and [8] focuses on generating accessible legal explanations. HyPA-RAG [9] is a hybrid system combining sparse (BM25) and dense (vector) methods with a knowledge graph retriever [10] for legal and policy applications.

For legal document drafting, LexDrafter [11] generates *Definition* articles for EUR-Lex, while CLERC [12] provides a dataset for retrieving and generating legal citations.

Legal Document Building Legal document building (or assembly) involves structured text generation using Knowledge Bases [13], LLMs [1], or a combination of the two.

Text Segmentation (TS) is a key component, partitioning documents into coherent sections based on semantic, structural, or functional criteria. Methods range from linear (that predict segments in a sequence) [14] to hierarchical (that recursively split the input document) [15] and from region-oriented (that predict segment boundaries) [16] to class-oriented approaches (that predict segment classes given the linguistic unit, e.g. sentences) [17]. Functional TS, crucial for isolating argumentative sections, has been successfully applied using Conditional Random Fields (CRFs) [17, 18], statistical models that combine good performance with exceptionally low complexity.

Legal Information Retrieval ensures relevance in automated document building [19]. In this area, approaches evolved from rule-based systems [20] and ontologies [21] to LLM-driven approaches [22], enhancing tasks like named entity recognition [23] and case law retrieval [24].

3. The JusBuild architecture

Legal document building is a complex process requiring careful structuring, research, and refinement. Traditionally, it involves three key phases: (i) defining the document format, (ii) drafting content for each section, and (iii) editing. While human expertise remains essential, AI-driven tools can significantly enhance efficiency by providing direct support with template-based structuring, semantic document retrieval, and automated text generation.

Judicial decisions, particularly those requiring detailed reasoning, exemplify the intricacies of legal drafting. The process is influenced by interpretative frameworks, legal precedents, and subjective factors, including a judge's perspective, legal realism, and social contexts. A legal document builder must, therefore, maintain human oversight while leveraging AI to streamline drafting. In this perspective, integrating AI into document generation fosters structure and readability of legal texts, improving both efficiency and accessibility.

JusBuild is designed as a document builder architecture to assist judges in drafting court orders and decisions while preserving their autonomy. A key design principle of JusBuild is the *human-in-the-loop* approach, ensuring that the judge retains full control over the drafting process. Users can either select suggested textual content or refine it based on legal and factual considerations. The JusBuild architecture is illustrated in Figure 1.

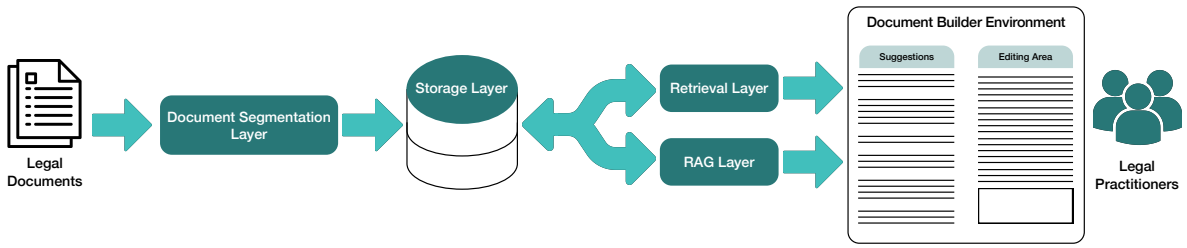


Figure 1: The JusBuild architecture for legal document drafting.

JusBuild employs AI-driven techniques to facilitate the document assembly process. It relies on a predefined legal document template and a structured corpus of legal documents, which are systematically segmented into sections, which are in turn indexed in a database. These data serve as the foundation for retrieving and generating section-specific suggestions to support the drafting process.

The *Document Builder Environment* consists of two primary components: an *editing area*, where users draft sections of a legal document, and a *suggestion area*, which provides relevant text recommendations that can be reused or modified. The drafting process follows a structured template, composed of sections $x_1, \ldots, x_M$. Here, we define d^* as the *active document* currently being edited and x^* as the *active section* under revision. Each section x_m $(m = 1, \ldots, M)$ is assigned a section label l_m from a predefined set of labels K.

JusBuild is characterized by the following featuring functionalities:

- A **Document Segmentation Layer**, based on the model proposed in [18], which automatically partitions legal documents into functional sections aligned with the predefined template using supervised classification techniques.
- A **Retrieval-Augmented Generation (RAG) Layer**, which expands the set of textual suggestions retrieved from the legal document corpus by incorporating AI-generated content. This hybrid approach enriches the drafting process by blending relevant judicial precedents with AI-generated text, offering judges a broader range of suggestions. The RAG pipeline ensures that generated content remains contextually relevant, aligns with past legal decisions, and maintains consistency with prior sections of the document.

To assist in drafting the active section, JusBuild provides two types of suggestions:

- The **Retrieved Sections**, denoted as $\hat{x}_M^1, \ldots, \hat{x}_M^I$, are obtained by retrieving sections labeled l_M from past documents in which the preceding sections $(l_1, \ldots, l_{M-1})$ closely resemble $x_1, \ldots, x_{M-1}$.
- The **Generated Sections**, represented as $\tilde{x}_M^1, \ldots, \tilde{x}_M^J$, are produced by a generative large language model (LLM). These are created using an analogy-based prompt that leverages the Retrieved Sections to generate new, contextually relevant content.

Suggestions are extracted from a corpus $\mathcal{C}$ of legal documents (e.g., court decisions), each denoted as z. Individual sections within these documents are referred to as z_k, and sentences within sections are labeled as s. Embedding vectors for sections—whether from the active document or from $\mathcal{C}$—as well as for sentences, are represented using boldface notation.

At initialization, the corpus $\mathcal{C}$ undergoes processing by the **Document Segmentation Layer**, which applies text segmentation based on sentence embeddings $\mathbf{s} \in \mathbb{R}^D$. The segmented sections, along with their corresponding Section Embeddings $\mathbf{z}_k \in \mathbb{R}^E$, computed using a dedicated embedding model, are then stored in the **Storage Layer**. This layer leverages a Vector Database to facilitate efficient retrieval within the Document Builder Environment.

During query execution, the Storage Layer interacts with both the **Document Retrieval Layer** and the **RAG Layer**, which respectively return the *Retrieved Sections* $\hat{x}_M^1, \ldots, \hat{x}_M^I$ and the *Generated Sections* $\tilde{x}_M^1, \ldots, \tilde{x}_M^J$, providing the user with a comprehensive set of suggestions for drafting the active section.

4. The JusBuild layers

The JusBuild architecture is composed of four main layers, namely the *Document Segmentation Layer*, the *Storage Layer*, the *Document Retrieval Layer*, and the *RAG Layer* described in the following.

4.1. Document Segmentation Layer

The Document Segmentation Layer partitions legal documents into *functionally coherent* sections, recognizing that judicial texts serve distinct roles such as `Introduction`, `Argumentation`, and `Conclusions`. Many older documents lack standardized formatting, making automated segmentation essential.

We frame functional Text Segmentation as a supervised classification task at the sentence level, assuming each sentence serves a single function. However, this approach presents challenges, including data scarcity (due to the need for expert annotation), label imbalance (as sections vary in length), legal-specific jargon, and potential discontinuities in section structure.

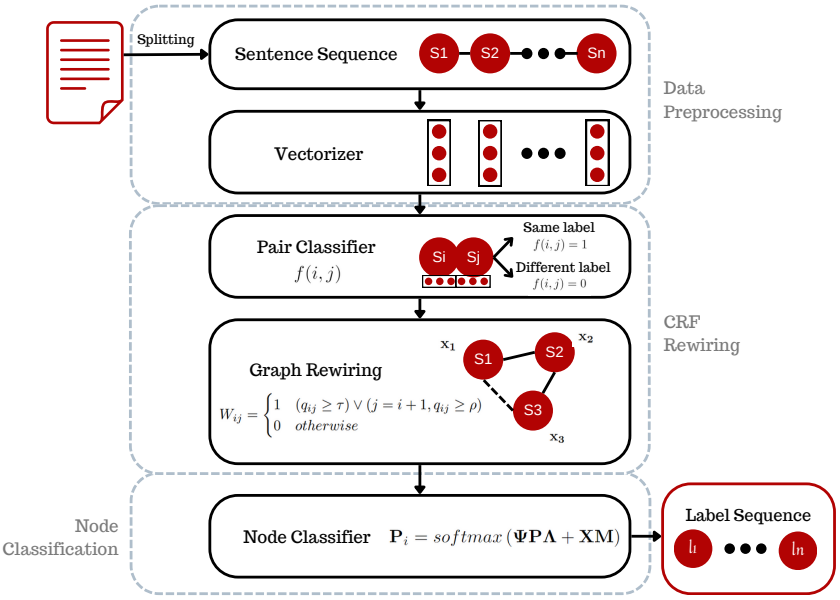


Figure 2: Few-shot functional Text Segmentation model.

To address these issues, Ferrara et al. [18] proposed a Conditional Random Field (CRF)-based model (Figure 2) that enhances traditional segmentation by incorporating an auxiliary Pair Classifier. This

classifier estimates the likelihood q_{ij} that two sentences (s_i, s_j) belong to the same section, modifying the CRF's graph structure: adding links for $q_{ij} \geq \tau$ and pruning for $q_{ij} < \rho$ (with thresholds $\rho < \tau$). The final classification is obtained using a CRF model:

$$\mathbf{P}_i = softmax(\Psi \mathbf{P} \Lambda + \mathbf{S}\mathbf{M}) \tag{1}$$

where $\mathbf{P}_i \in \Sigma^K$ (Σ^K being the simplex of dimension K) is the vector of probabilities of each section label for sentence s_i, $\Psi \in [0;1]^{N \times N}$ denotes the transition matrix of the rewired graph, $\mathbf{S} \in \mathbb{R}^{N \times D}$ is the feature matrix of all sentence nodes in the graph, i.e. the matrix of sentence embeddings, and $\Lambda \in \mathbb{R}^{K \times K}$ and $\mathbf{M} \in \mathbb{R}^{D \times K}$ are weight matrices to be learnt.

The model improves segmentation accuracy through:

- **Pairwise classification**, leveraging sentence pairs ($\binom{N}{2}$ per document) to combat data scarcity and enable few-shot segmentation;
- **Graph rewiring**, handling section discontinuities common in judicial texts;
- **Domain-specific embedding** and/or **Feature engineering**, ensuring sound legal text representation;
- **Loss weighting**, emphasizing rare section labels to mitigate class imbalance;
- **Positional knowledge injection**, initializing $\mathbf{P}$ to enforce known section order (e.g., `Introduction` first, `Conclusions` last).

Labeled sentences are finally grouped to reconstruct the full sections.

4.2. Storage Layer

The Storage Layer manages Document Sections and their embeddings, supporting CRUD operations. For each section z_k, it stores its document ID, position, text, section label l_{z_k}, and embedding vector $\mathbf{z}_k \in \mathbb{R}^E$.

To facilitate section-level retrieval, we employ a specialized embedding model designed for long-context processing using LSG (Local, Sparse, Global) attention [25]. This enables capturing section semantics over tens of thousands of tokens, reducing them to a single vector of dimension $E \approx 10^3$.

Efficient querying is enabled by a Vector Database, which supports fast similarity searches based on a predefined distance metric δ. This is critical for both Document Retrieval and RAG Layers, ensuring relevant sections from past documents are retrieved for reference.

4.3. Document Retrieval Layer

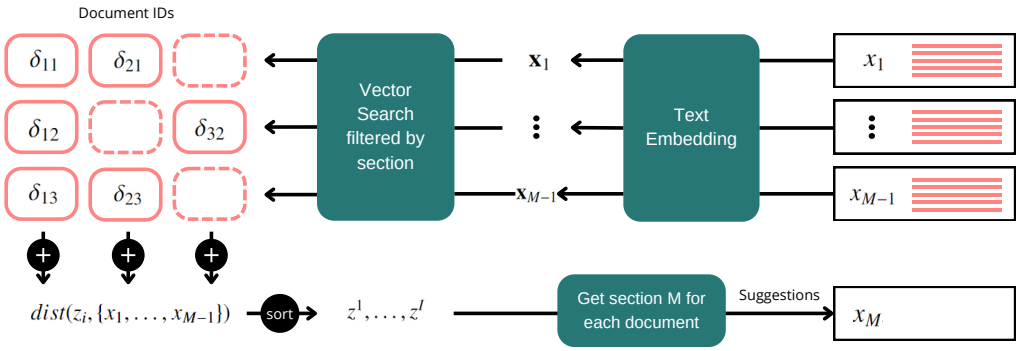


Figure 3: Document Retrieval Layer architecture.

The Document Retrieval Layer (Figure 3) identifies stored documents $z^1, \ldots, z^I$ whose sections $z^i_{l_1}, \ldots, z^i_{l_{M-1}}$ are similar to the active document's written sections $x_1, \ldots, x_{M-1}$. Each section is independently searched in parallel, as section order may vary.

The process involves embedding $x_1, \ldots, x_{M-1}$ into vectors $\mathbf{x}_1, \ldots, \mathbf{x}_{M-1} \in \mathbb{R}^E$ and querying the Storage Layer. For each section label l_m, the database returns the I closest stored sections:

$$z^1_{l_m} = \arg \min_{z_k : l_k = l_m} \delta(\mathbf{x}_m, \mathbf{z}_k). \tag{2}$$

To rank entire documents, we aggregate section distances. Given distance δ_{im} between x_m and $z^i_{l_m}$, we sum distances for each document, penalizing missing sections with the I-th closest distance:

$$dist(z_i, \{x_1, \ldots, x_{M-1}\}) = \sum_{m=1}^{M-1} \hat{\delta}_{im}, \quad \hat{\delta}_{im} = \begin{cases} \delta_{im}, & \text{if } l_m \in L^i, \\ \delta_{Im}, & \text{otherwise.} \end{cases} \tag{3}$$

This approach efficiently approximates true section distances while leveraging the vector database's computational power. The top I closest documents are selected, and their l_M sections (e.g., Conclusions) are extracted as $\hat{x}^1_M, \ldots, \hat{x}^I_M$ for downstream processing in the Document Builder Environment and RAG Layer.

4.4. RAG Layer

The RAG Layer expands the set of suggested sections for the user by leveraging a generative LLM to produce new section candidates $\tilde{x}^1_M, \ldots, \tilde{x}^J_M$. To ensure coherence with the draft document d^*—i.e., its existing sections $x_1, \ldots, x_{M-1}$—and align with patterns observed in similar cases, we employ a tailored prompting strategy.

At the core of this approach is *one-shot learning*, where the most similar retrieved document z^1 (excluding its final section $\hat{x}^1_M$) is presented as an example. This balances context relevance and prompt efficiency, avoiding the computational overhead of multi-shot prompts. The LLM receives the following structured prompt:

> *Given this document:* z^1. *Given its next section:* $\hat{x}^1_M$. *Now, generate the next section for this new document:* $x_1, \ldots, x_{M-1}$.

Repeating this process J times yields multiple *Generated Sections*, $\tilde{x}^1_M, \ldots, \tilde{x}^J_M$. These are combined with the *Retrieved Sections* and presented to the user in the Document Builder Environment, offering flexible drafting options.

5. Experimental results

To ensure robustness across diverse legal applications, we validate our architecture on two distinct datasets, differing in language, document structure, and legal domain. The evaluation focuses on generating the final court decision section, assessing three key components: Document Segmentation, Document Retrieval, and RAG.

For segmentation, we benchmark our approach against a linear-chain CRF, a common baseline for text segmentation. In retrieval, we measure the similarity between retrieved and actual conclusions using ROUGE-L (favouring recall) and BLEURT (favouring precision), alongside the Mean Reciprocal Rank (MRR) to evaluate ranking accuracy. For the RAG Layer, we conduct three comparisons using these similarity metrics: (i) different LLMs, (ii) retrieved vs. generated conclusions, and (iii) conclusions generated with vs. without the RAG pipeline.

5.1. Data and Setup

The first dataset [17] consists of 173 U.S. trade secret court decisions annotated with functional segments, while the second [26] includes 38 Italian unfair competition rulings labeled by legal experts. Despite their differences, both datasets follow a structured format, with text segmentation based on predefined

section labels. The evaluation considers entire sections for retrieval and generation, focusing on the concluding segment.

Experiments are run on a Zorin 16.3 machine (32GB RAM, 16-core CPU). Some architectural features remain metadata-independent (e.g., segmentation hyperparameters, retrieval count), while others depend on document metadata (e.g., language-specific embeddings and prompts). The following is a full list of the hyperparameter configurations:

- *Segmentation:* Uses a feedforward neural network (256 hidden units, ReLU activation), trained with AdamW and cyclic learning rates $[10^{-3}, 10^{-2}]$. We test $\tau \in \{0.65, 0.80, 0.95\}$ and $\rho = \tau/2$. Sentence embedding employs Legal-BERT for English and Italian-Legal-BERT for Italian texts.
- *Storage and Retrieval:* Chroma DBMS, L^2 distance metric, and HNSW indexing. LSG-BART provides section embeddings. We retrieve $I = 10$ sections, targeting "Conclusions" (English) or "Decision" (Italian).
- *RAG:* We generate $J = 1$ section per request, comparing GPT-4o-Mini and Mistral-7B [27] to evaluate performance.

5.2. Results

We evaluate the performance of segmentation, retrieval, and generation separately.

Document Segmentation. Our approach is benchmarked against a linear-chain CRF, with F_1 scores reported in Table 1. While our model surpasses the baseline on Italian data, it underperforms on US cases, possibly due to class imbalance (e.g., frequent "Analysis" sections) and limitations in sentence vectorization. Notably, our model remains consistent across different label sets, suggesting robustness despite data scarcity.

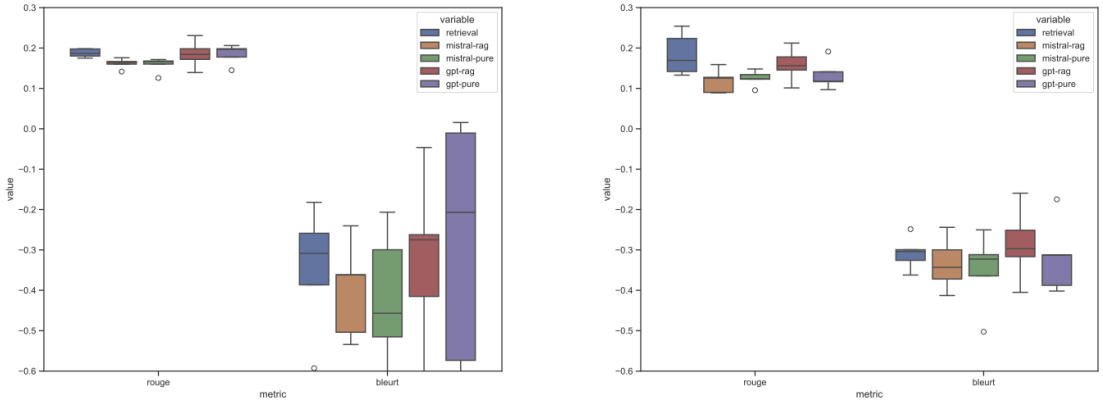
Model	Italian	US (5 sections)	US (7 sections)
Baseline (linear-chain CRF)	0.622	**0.717**	-
Ours ($\tau = 0.65$)	0.620	0.556	*0.556*
Ours ($\tau = 0.80$)	*0.627*	0.558	0.554
Ours ($\tau = 0.95$)	**0.650**	*0.563*	**0.561**

Table 1
F_1 scores for segmentation on Italian and US datasets. Bold indicates the best score, italics the second best.

Document Retrieval. Retrieval is assessed via Mean Reciprocal Rank (MRR) based on ROUGE-L and BLEURT scores. For Italian cases, the top-ranked retrieved conclusion averages between the first and second position ($MRR_{ROUGE-L} = 0.556$, $MRR_{BLEURT} = 0.567$). For US cases, retrieval is more challenging due to substantial variation in "Conclusions" sections, reflected in lower scores ($MRR_{ROUGE-L} = 0.357$, $MRR_{BLEURT} = 0.517$). The reliance of ROUGE-L on exact word sequences contributes to this drop.

RAG Performance. We compare two LLMs (Mistral-7B and GPT-4o-Mini) in generating conclusions, evaluating retrieval vs. generation and the impact of prompt augmentation. Figure 4 shows that GPT-4o-Mini consistently outperforms Mistral-7B, though the latter benefits from being open-source and locally executable. Generated conclusions often match or surpass retrieved ones in quality, suggesting their viability for legal drafting. Prompt augmentation aids generation in Italian cases, especially for GPT-4o-Mini, likely due to its larger context length. However, results for US cases remain ambiguous, with increased variance in pure generation scenarios.

In analysing the generated text, we find that RAG-produced conclusions often encapsulate key legal determinations, aligning with the true decisions despite differences in specificity. While human



(a) (b)

Figure 4: ROUGE-L and BLEURT scores for generated conclusions (Mistral-7B in green/orange, GPT-4o-Mini in purple/red) vs. best retrievals (blue) for US (a) and Italian (b) data.

refinement remains necessary, the system effectively supports legal drafting by providing structured, relevant content.

6. Concluding remarks

We present JusBuild, an NLP-powered architecture designed to assist legal practitioners in drafting case law decisions. JusBuild integrates: i) a CRF-based segmentation model to structure legal texts into functional sections, ii) a vector database for efficient semantic search, and iii) a hybrid retrieval and generation system suggesting precedent-based and LLM-generated content, always maintaining a human-in-the-loop approach.

Future work includes fine-tuning LLMs with legal-specific data, leveraging user feedback for model improvement, and exploring tailored prompting techniques to enhance document generation.

Acknowledgments

This work is supported in part by PNRR-NGEU program under MUR 118/2023, and in part by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU. Views and opinions expressed are those of the authors only, and do not necessarily reflect those of the European Union or the Italian MUR. Neither the European Union nor the Italian MUR can be held responsible for them.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT, Claude in order to: Grammar and spelling check, Paraphrase and reword. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] S. Castano, A. Ferrara, S. Montanelli, S. Picascia, D. Riva, A knowlegde-based service architecture for legal document building, in: 2nd International Workshop on Knowledge Management and Process Mining for Law, volume 3637, Sherbrooke, Quebec, Canada, 2023, pp. 1–15.

[2] M. Buffa, A. Ferrara, S. Picascia, D. Riva, S. Castano, Enhancing legal document building with retrieval-augmented generation, Submitted to: Computer Law and Security Review (2025).

[3] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021. URL: https://arxiv.org/abs/2005.11401. arXiv:2005.11401.

[4] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, H. Wang, Retrieval-augmented generation for large language models: A survey, 2024. URL: https://arxiv.org/abs/2312.10997. arXiv:2312.10997.

[5] N. Wiratunga, R. Abeyratne, L. Jayawardena, K. Martin, S. Massie, I. Nkisi-Orji, R. Weerasinghe, A. Liret, B. Fleisch, Cbr-rag: Case-based reasoning for retrieval augmented generation in llms for legal question answering, in: J. A. Recio-Garcia, M. G. Orozco-del Castillo, D. Bridge (Eds.), Case-Based Reasoning Research and Development, Springer Nature Switzerland, Cham, 2024, pp. 445–460.

[6] R. Yang, Casegpt: a case reasoning framework based on language models and retrieval-augmented generation, 2024. URL: https://arxiv.org/abs/2407.07913. arXiv:2407.07913.

[7] J. Cui, M. Ning, Z. Li, B. Chen, Y. Yan, H. Li, B. Ling, Y. Tian, L. Yuan, Chatlaw: A multi-agent collaborative legal assistant with knowledge graph enhanced mixture-of-experts large language model, 2024. URL: https://arxiv.org/abs/2306.16092. arXiv:2306.16092.

[8] A. Louis, G. van Dijck, G. Spanakis, Interpretable long-form legal question answering with retrieval-augmented large language models, Proceedings of the AAAI Conference on Artificial Intelligence 38 (2024) 22266–22275. URL: https://ojs.aaai.org/index.php/AAAI/article/view/30232. doi:10.1609/aaai.v38i20.30232.

[9] R. Kalra, Z. Wu, A. Gulley, A. Hilliard, X. Guan, A. Koshiyama, P. C. Treleaven, HyPA-RAG: A hybrid parameter adaptive retrieval-augmented generation system for AI legal and policy applications, in: S. Kumar, V. Balachandran, C. Y. Park, W. Shi, S. A. Hayati, Y. Tsvetkov, N. Smith, H. Hajishirzi, D. Kang, D. Jurgens (Eds.), Proceedings of the 1st Workshop on Customizable NLP: Progress and Challenges in Customizing NLP for a Domain, Application, Group, or Individual (CustomNLP4U), Association for Computational Linguistics, Miami, Florida, USA, 2024, pp. 237–256. URL: https://aclanthology.org/2024.customnlp4u-1.18/. doi:10.18653/v1/2024.customnlp4u-1.18.

[10] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, D. Metropolitansky, R. O. Ness, J. Larson, From local to global: A graph rag approach to query-focused summarization, 2025. URL: https://arxiv.org/abs/2404.16130. arXiv:2404.16130.

[11] A. Chouhan, M. Gertz, LexDrafter: Terminology drafting for legislative documents using retrieval augmented generation, in: N. Calzolari, M.-Y. Kan, V. Hoste, A. Lenci, S. Sakti, N. Xue (Eds.), Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), ELRA and ICCL, Torino, Italia, 2024, pp. 10448–10458. URL: https://aclanthology.org/2024.lrec-main.913.

[12] A. B. Hou, O. Weller, G. Qin, E. Yang, D. Lawrie, N. Holzenberger, A. Blair-Stanek, B. V. Durme, Clerc: A dataset for legal case retrieval and retrieval-augmented analysis generation, 2024. URL: https://arxiv.org/abs/2406.17186. arXiv:2406.17186.

[13] M. Marković, S. Gostojić, Legal document assembly system for introducing law students with legal drafting, Artificial Intelligence and Law 31 (2022) 829–863. URL: https://doi.org/10.1007/s10506-022-09339-2. doi:10.1007/s10506-022-09339-2.

[14] O. Koshorek, A. Cohen, N. Mor, M. Rotman, J. Berant, Text segmentation as a supervised learning task, in: M. Walker, H. Ji, A. Stent (Eds.), Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 469–473. URL: https://aclanthology.org/N18-2075. doi:10.18653/v1/N18-2075.

[15] G. Glavaš, A. Ganesh, S. Somasundaran, Training and domain adaptation for supervised text segmentation, in: J. Burstein, A. Horbach, E. Kochmar, R. Laarmann-Quante, C. Leacock, N. Madnani, I. Pilán, H. Yannakoudakis, T. Zesch (Eds.), Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications, Association for Computational Linguistics, Online,

2021, pp. 110–116. URL: https://aclanthology.org/2021.bea-1.11.
[16] A. Solbiati, K. Heffernan, G. Damaskinos, S. Poddar, S. Modi, J. Cali, Unsupervised topic segmentation of meetings with bert embeddings, 2021. URL: https://arxiv.org/abs/2106.12978. arXiv:2106.12978.
[17] K. D. Savelka, Jaromir; Ashley, Segmenting us court decisions into functional and issue specific parts., in: JURIX, 2018, pp. 111–120.
[18] A. Ferrara, S. Picascia, D. Riva, Few-shot legal text segmentation via rewiring conditional random fields: A preliminary study, in: International Conference on Conceptual Modeling, Springer, 2023, pp. 141–150.
[19] C. Sansone, G. Sperlí, Legal information retrieval systems: State-of-the-art and open issues, Information Systems 106 (2022) 101967. URL: https://www.sciencedirect.com/science/article/pii/S0306437921001551. doi:https://doi.org/10.1016/j.is.2021.101967.
[20] W. Y. Mok, J. R. Mok, Legal machine-learning analysis: First steps towards a.i. assisted legal research, in: Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law, ICAIL '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 266–267. doi:10.1145/3322640.3326737.
[21] S. Castano, A. Ferrara, M. Falduti, S. Montanelli, Crime knowledge extraction: An ontology-driven approach for detecting abstract terms in case law decisions, in: Proc. of the 17th Int Conference on Artificial Intelligence and Law, ICAIL '19, ACM, New York, NY, USA, 2019, p. 179–183. doi:10.1145/3322640.3326730.
[22] A. Ferrara, S. Picascia, D. Riva, Context-Aware Knowledge Extraction from Legal Documents through Zero-Shot Classification, in: Proc. of the 1st ER Int. Workshop on Digital Justice, Digital Law, and Conceptual Modeling (JUSMOD22), Hyderabad, India, 2022, p. 81 – 90.
[23] A. Elnaggar, R. Otto, F. Matthes, Deep learning for named-entity linking with transfer learning for legal documents, in: Proceedings of the 2018 Artificial Intelligence and Cloud Computing Conference, AICCC '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 23–28. URL: https://doi.org/10.1145/3299819.3299846. doi:10.1145/3299819.3299846.
[24] Y. Shao, J. Mao, Y. Liu, W. Ma, K. Satoh, M. Zhang, S. Ma, Bert-pli: Modeling paragraph-level interactions for legal case retrieval, in: C. Bessiere (Ed.), Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, International Joint Conferences on Artificial Intelligence Organization, 2020, pp. 3501–3507. URL: https://doi.org/10.24963/ijcai.2020/484. doi:10.24963/ijcai.2020/484, main track.
[25] C. Condevaux, S. Harispe, Lsg attention: Extrapolation of pretrained transformers to long sequences, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2023, pp. 443–454.
[26] E. Zanoli, M. Barbini, D. Riva, S. Picascia, E. Furiosi, S. D'Ancona, C. Chesi, et al., Annotators-in-the-loop: testing a novel annotation procedure on italian case law, in: Proceedings of the 17th Linguistic Annotation Workshop (LAW-XVII), Association for Computational Linguistics, 2023, pp. 118–128.
[27] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al., Mistral 7b, arXiv preprint arXiv:2310.06825 (2023).