# LeXIDesk: Legal AI Workbench

Project Phase I Report

*submitted by*

**MALAVIKA SURESH**

Reg. No. MAC22CS042

**SANIKA SIVA S**

Reg. No. MAC22CS054

**SHIVANI SHIBU**

Reg. No. MAC22CS056

*to*

**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**

*in partial fulfilment of the requirements for the award of the Degree*

*of*

**BACHELOR OF TECHNOLOGY**

*in*

**COMPUTER SCIENCE AND ENGINEERING**


**Department of Computer Science & Engineering**

**Mar Athanasius College of Engineering ( Autonomous )**

**Kothamangalam, Kerala, India 686 666**

NOVEMBER 2025

# LeXIDesk: Legal AI Workbench

Project Phase I Report

*submitted by*

**MALAVIKA SURESH**

Reg. No. MAC22CS042

**SANIKA SIVA S**

Reg. No. MAC22CS054

**SHIVANI SHIBU**

Reg. No. MAC22CS056

*to*

**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**

*in partial fulfilment of the requirements for the award of the Degree*

*of*

**BACHELOR OF TECHNOLOGY**

*in*

**COMPUTER SCIENCE AND ENGINEERING**


**Department of Computer Science & Engineering**

**Mar Athanasius College of Engineering ( Autonomous )**

**Kothamangalam, Kerala, India 686 666**

NOVEMBER 2025

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**MAR ATHANASIUS COLLEGE OF ENGINEERING**

**( AUTONOMOUS )**

**KOTHAMANGALAM**

## CERTIFICATE

*This is to certify that the report entitled* **LeXIDesk: Legal AI Workbench** *submitted by* **MALAVIKA SURESH (Reg No. MAC22CS042), SANIKA SIVA S (Reg No. MAC22CS054), SHIVANI SHIBU (Reg No. MAC22CS056)**, *towards partial fulfillment of the requirement for the award of Degree of Bachelor of Technology in Computer Science and Engineering from APJ Abdul Kalam Technological University for November 2025 is a bonafide record of the project carried out by them under our supervision and guidance.*

**Prof. Richu Shibu**
Project Guide

**Prof. Dr. Elizabeth Issac**
Project Guide

**Prof. Richu Shibu**
Project Coordinator

**Prof. Joby George**
Head of the Department

Date:

Dept. Seal

# ACKNOWLEDGEMENT

*First and foremost, we sincerely thank the 'God Almighty' for his grace for the successful and timely completion of the project.*

*We express our sincere gratitude and thanks to* ***Dr. Bos Mathew Jos****, Principal and* ***Prof. Joby George****, Head of the Department for providing the necessary facilities and their encouragement and support.*

*We owe special thanks to our project guide* ***Prof. Dr. Elizabeth Issac*** *and project coordinator and guide* ***Prof. Richu Shibu*** *for their corrections, suggestions and sincere efforts to coordinate the project under a tight schedule.*

*We express our sincere thanks to staff members in the Department of Computer Science and Engineering who have taken sincere efforts in guiding and correcting us in conducting this project.*

*Finally, we would like to acknowledge the heartfelt efforts, comments, criticisms, co-operation and tremendous support given to us by our dear friends during the preparation of the project and also during the presentation without which this work would have been all the more difficult to accomplish.*

# ABSTRACT

The Legal AI Workbench addresses inefficiencies in legal workflows caused by manual, fragmented processes in clause review, contract negotiation, and litigation research. It offers an advanced AI-driven platform for automating and enhancing legal document processing using sophisticated Natural Language Processing (NLP) methods. A core component is robust sentence boundary detection (SBD) tailored for legal texts. Leveraging a hybrid deep learning architecture combining Convolutional Neural Networks (CNN) and Conditional Random Fields (CRF), the model captures intricate character-level contexts around delimiters while modeling sequential token dependencies. Legal documents present challenges such as complex structures, frequent citations, abbreviations, and irregular punctuation. The hybrid CNN-CRF model achieves a 4% F1-score improvement over baselines, and outperforming standard NLP tools by 8% on out-of-domain Indian legal datasets. This precise segmentation provides a vital preprocessing foundation for downstream tasks like clause segmentation, adaptive negotiation support, and predictive litigation analytics. The system ultimately reduces review time, uncovers hidden risks, and promotes consistent, explainable legal decision-making within an integrated legal assistant framework.

# Table of Contents

# List of Figures

# List of Abbreviations

| | |
|---|---|
| CRF | Conditional Random Field |
| SBD | Sentence Boundary Detection |
| LSTM | Long Short-Term Memory |
| BiLSTM | Bidirectional Long Short-Term Memory |
| PyTorch | Python Deep Learning Framework |
| NLTK | Natural Language Toolkit |
| SpaCy | Python NLP Library |
| NER | Named Entity Recognition |
| BERT | Bidirectional Encoder Representations from Transformers |
| XLNet | Generalized Autoregressive Pretraining for Language Understanding |
| BLEU | Bilingual Evaluation Understudy |
| TF-IDF | Term Frequency-Inverse Document Frequency |
| RAG | Retrieval-Augmented Generation |
| PEFT | Parameter-Efficient Fine-Tuning |
| LoRA | Low-Rank Adaptation |
| GRU | Gated Recurrent Unit |

# Chapter 1

# Introduction

In recent years, Natural Language Processing (NLP) has become a cornerstone technology in advancing the automation of legal document analysis, transforming the way legal professionals manage, interpret, and process large volumes of unstructured text. Tasks such as case retrieval, argument extraction, and document summarization rely fundamentally on precise linguistic structuring, of which Sentence Boundary Detection (SBD) forms the base. Reliable boundary detection is essential, as inaccuracies at this stage can propagate through subsequent NLP pipelines—degrading performance in summarization, classification, and search applications. While humans can intuitively recognize where one sentence ends and another begins, replicating this process algorithmically remains challenging, particularly in the legal domain.

Legal texts differ significantly from general English corpora due to their complex syntactic structures and domain-specific expressions. Typically, they contain lengthy, embedded clauses, citations, enumerations, and abbreviations that complicate automated parsing. Traditional rule-based SBD methods often fail under these conditions. For instance, a period occurring within abbreviations or legal citations does not necessarily signify the end of a sentence; conversely, colons or semicolons may serve as sentence boundaries. Consequently, improving contextual understanding of punctuation and structure is essential for building models that perform reliably in legal contexts.

To overcome these issues, this study proposes a hybrid deep learning framework that merges statistical and neural modeling techniques for sentence boundary detection in legal texts. The system combines a Convolutional Neural Net-

work (CNN) for capturing local, character-level contextual patterns surrounding potential boundary tokens, with a Conditional Random Field (CRF) model responsible for learning sequence-level dependencies. This dual-structured architecture leverages the representational power of neural networks together with the contextual reasoning ability of statistical modeling, yielding more accurate and context-sensitive boundary predictions.

For evaluation, the system was trained and tested on a diverse set of legal documents, including judgments from the Indian Supreme Court and adjudicatory decisions from the United States. Results demonstrate a consistent improvement in detection accuracy over conventional CRF-based and CNN-only architectures, underscoring the hybrid model's capacity to handle complex syntactic variations present in legal writing.

Accurate sentence segmentation not only enhances model performance for document analysis but also lays the foundation for more advanced applications such as legal summarization and analytics. Building upon the SBD component, this research will progress toward the creation of a domain-adaptive summarization framework capable of generating concise and context-aware summaries of lengthy judgments. Together, these modules—starting with precise sentence boundary recognition and extending to intelligent summarization—serve as integral components in developing an AI-powered legal workbench aimed at improving efficiency, transparency, and accessibility in legal processes.

# Chapter 2

# Background

## 2.1 Natural Language Processing (NLP)

Natural Language Processing (NLP) represents a multidisciplinary domain that combines computer science, linguistics, and artificial intelligence to enable machines to interpret and produce human language with semantic accuracy. Over time, it has become fundamental in diverse applications such as search systems, dialogue agents, translation, sentiment analysis, and text summarization. In the legal sector, NLP encounters greater complexity because of the intricate and highly formalized structure of legal discourse. Legal drafting typically involves long, nested clauses, numerous citations and enumerations, and specialized jargon that restrict straightforward syntactic parsing and semantic interpretation.

The advancement of deep learning has significantly improved NLP capabilities by allowing models to capture contextual relations and nuanced meanings within text. Hybrid models that integrate statistical and neural methods have proven particularly useful for handling domain-specific challenges where either approach individually performs inadequately. Here, NLP serves as the core technological layer supporting both the sentence segmentation and summarization components.

## 2.2 Sentence Boundary Detection (SBD)

Sentence Boundary Detection (SBD) is a foundational task in most NLP pipelines, tasked with locating precise sentence endings and beginnings within a text. Although humans recognize these transitions effortlessly, automating the

process, especially for legal documents, poses challenges due to the domain's unconventional use of punctuation. Traditional rule-based techniques generally depend on explicit delimiters such as periods, question marks, or exclamation marks; however, in the legal context, such assumptions often fail. Abbreviations like "U.S." or legal citations such as "AIR 2021 SC 45." illustrate that punctuation does not always mark a boundary.

Contemporary methods resolve these ambiguities by employing machine learning and deep learning architectures capable of learning contextual cues. Among these, hybrid CNN–CRF frameworks have demonstrated exceptional effectiveness. The CNN component captures fine-grained character-level patterns surrounding delimiters, while the CRF models sequential dependencies across tokens. Their combination enhances segmentation accuracy in complex legal texts where rules alone cannot suffice.

## 2.3 Text Summarization

Text summarization condenses lengthy documents into shorter versions while retaining core meanings and essential arguments. It generally falls into two categories: extractive summarization, which compiles key sentences or phrases from the source, and abstractive summarization, which generates new text that paraphrases the original. For legal practitioners, summarization offers substantial value by reducing the time needed to review voluminous judgments, case laws, and contracts. Generic summarization tools tend to fail in this setting because they neglect the logical constructs and reasoning patterns unique to legal writing.

Accurate sentence segmentation significantly enhances summarization outcomes by supplying well-structured and contextually coherent input. Consequently, this research pursues a domain-focused summarization mechanism trained on legal corpora that builds upon robust sentence boundary detection to achieve context-aware condensation.

## 2.4 Deep Learning in NLP

Deep learning methodologies have transformed NLP, surpassing traditional statistical approaches such as Hidden Markov Models (HMMs) and standard CRFs. Neural architectures learn hierarchical and distributed text representations conducive to understanding intricate linguistic dependencies. CNNs are adept at identifying localized textual features like word or character sequences, whereas recurrent models such as LSTMs and GRUs capture temporal dependencies in text sequences. Transformer-based architectures, including BERT and GPT, have further advanced the field by efficiently modeling long-range context and achieving top-tier performance across numerous NLP benchmarks.

In the hybrid CNN–CRF configuration, CNN layers extract localized contextual signals around punctuation, which the CRF subsequently uses to model inter-sentence relationships. This synergy capitalizes on CNN's representational power and CRF's capability for structured sequence prediction, enabling robust and accurate sentence segmentation even within verbose legal narratives.

## 2.5 Python

Python serves as the principal programming language across all stages of the project. Known for its simplicity, readability, and comprehensive library ecosystem, Python supports multiple paradigms, from functional and procedural to object-oriented programming. Its extensive collection of frameworks for scientific computing, AI, and text processing—such as NumPy, Pandas, and PyTorch—makes it an optimal choice for implementing machine learning models. The language's flexibility, integration capacity, and active development community facilitate efficient experimentation, large-scale text data management, and rapid prototyping of neural and statistical models.

## 2.6 Libraries and Frameworks

### 2.6.1 PyTorch

PyTorch, an open-source framework developed by Meta AI, offers dynamic computational graphs that make model construction and modification intuitive. Its GPU acceleration and modular structure are highly suitable for developing deep learning architectures such as CNNs. Within this project, PyTorch forms the backbone for training and optimizing the CNN component of the hybrid system.

### 2.6.2 sklearn-crfsuite

The sklearn-crfsuite package provides a Pythonic interface to the CRFsuite library, enabling efficient modeling of Conditional Random Fields. It simplifies sequence labeling tasks like sentence segmentation, named entity recognition, and part-of-speech tagging. In this project, it supports the CRF and CNN–CRF hybrid modeling frameworks for accurate legal sentence boundary detection.

### 2.6.3 NumPy and Pandas

NumPy facilitates numerical computation through its robust array structures and vectorized operations, while Pandas offers advanced data manipulation features such as DataFrames for structured data processing. Together, they streamline data preprocessing, feature engineering, and performance evaluation workflows required for training the SBD model.

### 2.6.4 NLTK and SpaCy

Both NLTK (Natural Language Toolkit) and SpaCy are integral to natural language preprocessing. They assist in tokenization, lemmatization, and textual normalization, ultimately preparing the dataset for input into learning models. Their combined use ensures consistent, clean, and linguistically informed data for both statistical and deep learning pipelines.

## 2.7 Software Description

### 2.7.1 Visual Studio Code

Visual Studio Code (VS Code) serves as the core integrated development environment for coding, project management, and debugging. It offers features such as syntax highlighting, version control through Git integration, and a wide range of AI-focused extensions. These features streamline collaborative research and model experimentation, supporting reproducibility and team-based development throughout the project.

# Chapter 3

# Literature Review

The discipline of legal Natural Language Processing (NLP) has evolved considerably in recent years, with increasing attention directed toward sentence segmentation, automated summarization, and the development of intelligent conversational systems. Legal corpora present unique computational difficulties due to their dense syntax, specialized vocabulary, and unconventional punctuation styles, all of which complicate text interpretation and boundary detection. As legal writing often includes lengthy, nested clauses and complex referencing pat terns, designing models that can accurately process and structure such data remains a significant challenge. The key research contributions that explore advanced approaches to addressing these issues using deep learning, statistical-neural hybrid architectures, and transformer-based systems are highlighted. Collectively, these studies establish a strong foundation for the design and implementation of the proposed model, which seeks to improve the precision and efficiency of sentence boundary detection and subsequent legal text understanding tasks.

## 3.1 Legal SBD: Hybrid Deep Learning & Statistical Models

**Paper Title:** Legal sentence boundary detection using hybrid deep learning and statistical models

**Authors:** Reshma Sheik, Sneha Rao Ganta, S. Jaya Nirmala

**Source:** Artificial Intelligence and Law, Springer, 2024

Sentence boundary detection (SBD) is a crucial first step in natural language processing, especially challenging in legal text due to its complex structure and linguistic features. This paper proposes deep learning models including CNN, LSTM, GRU, BiLSTM (Bidirectional Long Short-Term Memory), and transformers such as LegalBERT and CaseLawBERT to leverage delimiter and surrounding context information for detecting sentence boundaries in English legal texts. The CNN model demonstrated best performance in F1 score, model size, and inference time. Further, integrating CNN features with a CRF model in a hybrid architecture yielded a 4% improvement in F1 score, outperforming baseline models and standard NLP toolkits on Indian legal datasets. This hybrid approach combines local feature extraction and sequential dependency modeling, improving robustness and accuracy in legal sentence segmentation [1].

### 3.1.1 Observations

- Deep learning effectively models sentence boundaries considering context and delimiters.
- CNN balances accuracy and computational efficiency better than other deep learning models.
- Hybrid CNN-CRF architecture outperforms standard CRF and standalone deep models.
- Generalizes well to Indian legal text and outperforms open-source NLP toolkits.
- Provides detailed evaluation across diverse legal datasets and token-level labeling.

### 3.1.2 Drawbacks

- Transformer models require larger datasets and have higher computational costs.
- CNN model relies mainly on period delimiter; other punctuation marks need further study.

- Some errors arise from unusual sentence structures and delimiter ambiguities.
- Limited exploration of rule-based or ensemble post-processing techniques.

## 3.2 Tibetan SBD: Context-Aware Bi-LSTM

**Paper Title:** ATibetan Sentence Boundary Disambiguation Model Considering the Components on Information on Both Sides of Shad
**Authors:** F. Li, H. Lv, Y. Gao, N. Dolha, Y. Li, Q. Zhou
**Source:** Tsinghua Science & Technology, 28(6), 2023

Sentence Boundary Disambiguation (SBD) is essential for accurate natural language processing and pretraining language models, especially for Tibetan, which has ambiguous punctuation marks like shad that can represent multiple sentence-ending functions. This study proposes a component-level Tibetan SBD method using a Bidirectional Long Short-Term Memory (Bi-LSTM) deep learning model that considers context on both sides of the shad punctuation. The approach reduces error amplification from word segmentation and part-of-speech tagging and demonstrates superior F1-score performance (96%) compared to other models. Experiments on low-resource languages such as Turkish and Romanian and high-resource languages like English and German further validate the model's generalization capabilities [10].

### 3.2.1 Observations

- Considers textual context on both sides of punctuation marks for improved disambiguation.
- Component-level analysis mitigates errors from segmentation and tagging.
- Bi-LSTM model outperforms CNN, LSTM, GRU, Bi-GRU, and MLP models in Tibetan SBD.
- Experimental validation includes multilingual datasets, showing broad generalizability.

- Window size significantly impacts model performance; an optimal window includes both left and right contexts.

### 3.2.2 Drawbacks

- High computational requirements due to deep learning and large context windows.
- Tibetan language-specific features may limit direct applicability to other languages.
- Performance sensitive to the size and quality of annotated training data.
- Complexity of Tibetan script and punctuation introduces unique annotation challenges.
- Future work needed to improve robustness for nonstandard and noisy data inputs.

## 3.3 LegalSeg: Rhetorical Role Classification

**Paper Title:** LegalSeg: Unlocking the Structure of Indian Legal Judgments Through Rhetorical Role Classification
**Authors:** S. K. Nigam, T. Dubey, G. Sharma, N. Shallum, K. Ghosh, A. Bhattacharya
**Source:** Findings of the Association for Computational Linguistics: NAACL 2025

LegalSeg is a large-scale dataset and benchmark designed for semantic segmentation of Indian legal judgments using Rhetorical Role (RR) Classification. Sentences are classified into seven roles: Facts, Issue, Arguments of Petitioner (AoP), Arguments of Respondent (AoR), Reasoning, Decision, and None. The dataset contains 7,120 documents and 1,487,149 sentences from the Supreme Court of India and various High Courts, addressing the historical lack of annotated corpora for training robust models. Several state-of-the-art models

were evaluated, including Hierarchical BiLSTM-CRF, TransformerOverInLegal-BERT (ToInLegalBERT), Graph Neural Networks (GNNs), InLegalBERT variants, and an instruction-tuned LLaMA model (RhetoricLLaMA). Context-aware and structure-aware models consistently outperformed sentence-only models. Hierarchical BiLSTM-CRF achieved the highest F1-score of 0.77 by modeling sequential dependencies and label transitions. ToInLegalBERT and GNNs also benefited from hierarchical and graph-based structures. Using predicted labels during inference sometimes improved robustness. RhetoricLLaMA showed potential but requires more domain-specific fine-tuning [2].

### 3.3.1 Observations

- Context and structural information significantly improve classification accuracy over sentence-only approaches.
- Hierarchical BiLSTM-CRF captures sequential dependencies and label transitions effectively.
- ToInLegalBERT and GNNs leverage document structure for better performance.
- Using predicted labels during inference improves model robustness.
- LegalSeg provides a large-scale benchmark to facilitate reproducibility and evaluation of legal NLP models.

### 3.3.2 Drawbacks

- Difficulty distinguishing closely related rhetorical roles (e.g., Facts vs. Reasoning, AoP vs. AoR).
- Class imbalance leads to overprediction of frequent labels (e.g., None, Facts).
- Instruction-tuned LLMs underperform without specialized domain adaptation.

- Cascading prediction errors affect overall segmentation accuracy.
- Dataset is specific to Indian judiciary; adaptation required for other jurisdictions.
- High computational requirements limit deployment in low-resource environments.

## 3.4 Deep Learning SBD in Legal Texts

**Paper Title:** Efficient Deep Learning-based Sentence Boundary Detection in Legal Text

**Authors:** R. Sheik, T. Gokul, S. Nirmala

**Source:** Proceedings of the Natural Legal Language Processing Workshop 2022

This study develops an efficient deep learning framework for Sentence Boundary Detection (SBD) tailored to complex legal text. Traditional SBD methods relying on punctuation and capitalization rules underperform in legal documents due to structural ambiguities such as long sentences, lists, citations, and varied use of periods. The proposed approach frames SBD as a binary classification problem using a context window-based deep learning architecture that leverages character-level context surrounding potential delimiters. Among tested models, the Convolutional Neural Network (CNN) achieved the best performance with an F1-score of 0.977, outperforming rule-based pySBD by 8% and transformer models (LEGAL-BERT, XLNet) in both efficiency and accuracy. Integrating CNN features with a CRF model yielded improved sequential modeling while maintaining high runtime efficiency, establishing CNN as the optimal choice for legal SBD [13].

### 3.4.1 Observations

- Legal texts present unique challenges for SBD, with only about 40% of periods indicating true sentence boundaries due to acronyms, initials, numbers, and citations.

- CNN models combine high performance (F1-score 0.977) with efficiency, small size (116 kB), and minimal trainable parameters (29,275), making them faster than CRF and transformer models.
- Transformers (LEGAL-BERT, XLNet) underperform in accuracy, require 40x longer training time, have high inference times (112-113 seconds), and large memory footprints ( 110M parameters).
- CNN-CRF hybrid improves sequential dependency modeling while retaining CNN efficiency.

### 3.4.2 Drawbacks

- The model focuses exclusively on periods (".") as sentence delimiters; other punctuation (e.g., colons, exclamation marks) are not classified due to insufficient data.
- Statistical CRF models can detect additional delimiters that CNN cannot.
- Complex cases, such as multiple periods in abbreviations or citations, remain challenging for both CNN and CRF.
- Future work requires extending to multiple delimiters and chaining models for enhanced SBD.
- Transformer models have high computational costs, requiring substantial GPU resources.

## 3.5 Where's the Point: Multilingual SBD

**Paper Title:** Where's the Point? Self-Supervised Multilingual Punctuation-Agnostic Sentence Segmentation
**Authors:** B. Minixhofer, J. Pfeiffer, I. Vulić
**Source:** Proceedings of the 61st ACL, 2023

Where's the Point (WtP) is a multilingual sentence segmentation framework designed to overcome the dependence of traditional tools on explicit punctuation

and large amounts of annotated data. It employs a bidirectional character-level language model (ChLM) trained in a fully self-supervised manner across 85 languages, treating paragraph breaks as implicit segmentation cues. This pragmatic approach enables segmentation in languages that lack clear sentence-ending punctuation, such as Thai. The adapted variant, WtPPUNCT, introduces an auxiliary punctuation-prediction objective that enhances corpus-specific adaptation using only 64–256 manually segmented sentences. WtP achieved an average 6.1% improvement in F1-score compared to prior methods and delivered an average gain of 2.3 Bilingual Evaluation Understudy(BLEU) points in machine translation tasks when segmentation aligned with MT training protocols. The model thus provides a robust, scalable solution for multilingual text processing without reliance on large labeled corpora [**?**].

### 3.5.1 Observations

- Ensures consistency between training and inference segmentation, crucial for improving downstream applications such as machine translation.
- Demonstrates strong adaptability, effectively segmenting languages that lack sentence-ending punctuation (e.g., Thai).
- The auxiliary punctuation-prediction objective benefits both segmentation accuracy and corpus adaptation.
- Threshold-based adaptation (WtPT) shows cross-lingual validity and minimal resource dependence.

### 3.5.2 Drawbacks

- Lower segmentation performance in certain low-resource languages (e.g., Welsh, Nepali, Punjabi, Pushto), possibly due to limitations in the mC4 pretraining corpus.
- Adapted WtPPUNCT classifiers exhibit weak transferability across different datasets or language collections.

- Performance may be biased toward languages and communities overrepresented in the training data.
- Computational cost increases for adaptation and fine-tuning in multilingual settings.

## 3.6 NUPunkt & CharBoundary

**Paper Title:** Precise Legal Sentence Boundary Detection for Retrieval at Scale: NUPunkt and CharBoundary
**Authors:** M. Bommarito, D. Katz, J. Bommarito
**Source:** arXiv preprint arXiv:2504.04131, 2025

NUPunkt and CharBoundary are two open-source sentence boundary detection (SBD) libraries optimized for high-precision and high-throughput processing of legal text. NUPunkt extends the unsupervised Punkt algorithm with legal domain-specific optimizations and achieves 91.1% precision while processing 10 million characters per second with modest memory usage. CharBoundary employs a character-level supervised machine learning approach with models in small, medium, and large sizes, offering balanced precision-recall tradeoffs and achieving a highest F1-score of 0.782. Evaluations on five diverse legal datasets with over 25,000 documents demonstrate that these approaches outperform general-purpose SBD tools by up to 29–32% in precision, significantly reducing fragmentation errors in legal NLP and retrieval-augmented generation applications [3].

### 3.6.1 Observations

- NUPunkt specializes in token-level unsupervised learning with a rich legal abbreviation dictionary and structural handling.
- CharBoundary applies character-level random forest models focusing on local character context and legal features.

- Both models achieve CPU-efficient high throughput suitable for scaling in large legal corpora.
- NUPunkt demonstrates exceptional precision, critical for retrieval-augmented legal applications.
- CharBoundary's models provide adaptable precision-recall balance supporting various deployment needs.

### 3.6.2 Drawbacks

- Limited language scope primarily to English legal documents.
- Adaptability to complex legal subdomains outside current datasets needs further validation.
- High computational demand for the CharBoundary large model.
- Some challenges remain in handling non-standard document formats and multilingual texts.
- Future improvements include multilingual extensions and hybrid modeling approaches.

## 3.7 Dynamic Legal RAG for Summarization

**Paper Title:** Optimizing Legal Text Summarization Through Dynamic Retrieval-Augmented Generation and Domain-Specific Adaptation
**Authors:** A. S. Mukund, K. S. Easwarakumar
**Source:** Symmetry, 17(5), 2025

This study proposes a novel framework for legal text summarization, integrating a Dynamic Legal Retrieval-Augmented Generation (RAG) system with domain-specific adaptation. Legal Named Entity Recognition (NER) identifies crucial entities, which act as query anchors for real-time retrieval from domain-specific repositories. The retrieval mechanism uses the BM25 algorithm

with top-3 chunk selection. The generative component fine-tunes decoder-only architectures, with LLaMA 3.1-8B achieving the best performance, yielding a BERTScore (F1) of 0.8906. Domain knowledge integration improves factual consistency (92.7%) and legal entity preservation (93.8%). A compression ratio constraint (0.05–0.5) ensures structural alignment between source judgments and summaries. This approach demonstrates that dynamic retrieval combined with generative models enhances accuracy, contextual relevance, and legal fidelity in automated summarization [4].

### 3.7.1 Observations

- Domain-specific adaptation is essential; general-purpose models often omit critical statutory references or hallucinate content.
- Sparse keyword-based BM25 retrieval outperforms dense methods (DPR, ColBERT, SGPT) for legal texts.
- Top-3 chunk retrieval balances sufficient context with minimal redundancy.
- Models with weaker baseline performance (e.g., LLaMA 2-7B) improve significantly with domain knowledge integration.
- Parameter-Efficient Fine-Tuning (PEFT) using LoRA (Low-Rank Adaptation) with NF4 quantization reduces memory overhead while maintaining performance.

### 3.7.2 Drawbacks

- Jurisdictional Limitation: Framework is specialized for Indian legal texts; adaptation to other jurisdictions requires extensive reengineering.
- Data Preprocessing: Strict filtering and compression ratio constraints led to 5.23% data loss to ensure consistency.
- High Computational Requirements: Models like LLaMA 3.1-8B require GPUs with 40 GB VRAM for efficient training and inference.

- Dense retrievers (DPR, ColBERT) have higher memory overhead and irrelevant retrieval rates compared to BM25.
- Evaluation Metrics: Standard metrics (ROUGE, BERTScore) inadequately capture legal and factual fidelity, highlighting the need for specialized evaluation.

## 3.8 Multilingual Legal SBD

**Paper Title:** MultiLegalSBD: A Multilingual Legal Sentence Boundary Detection Dataset

**Authors:** T. Brugger, M. Stürmer, J. Niklaus

**Source:** ICAIL '23, 2023

MultiLegalSBD introduces a large, diverse, high-quality multilingual legal dataset for Sentence Boundary Detection (SBD), containing over 130,000 annotated sentences across six languages: French, Spanish, Italian, English, German, and Portuguese (zero-shot data). The study evaluates Conditional Random Fields (CRF), BiLSTM-CRF, and transformer-based models (DistilBERT). Trained monolingual models achieved F1-scores in the high nineties, outperforming baselines. Multilingual transformer models performed comparably or better across all languages, reaching up to 99.2% F1-score. Zero-shot evaluation on Portuguese data demonstrated strong cross-lingual transfer, achieving F1-scores in the lower nineties and outperforming all baseline models. The study highlights the effectiveness of transformer-based architectures in capturing legal text complexities, including long sentences, complex structures, and ambiguous punctuation [11].

### 3.8.1 Observations

- Existing SBD systems (CoreNLP, NLTK, Spacy, Stanza) perform suboptimally on multilingual legal text.
- Legal texts present challenges such as long sentences, citations, parentheses,

lists, and ambiguous punctuation.

- Deep learning models (CRF, BiLSTM-CRF, transformers) achieve high F1-scores, comparable to curated news datasets.
- Transformers exhibit superior cross-domain and cross-lingual transfer, generalizing well between judgments and laws.

### 3.8.2 Drawbacks

- Dataset limited to Germanic and Italic language groups, potentially simplifying cross-lingual transfer.
- Single annotator limits annotation validation; additional native speakers could improve quality.
- CRF and BiLSTM-CRF models perform poorly in zero-shot transfer (78.6% and 73.2% F1 on Portuguese laws).
- Errors often arise from internal citations, parentheses, unknown abbreviations, and headline segmentation.
- Transformers have token length limits (512 tokens), complicating application to long, unseen texts.

## 3.9 Hybrid Transformer Summarization

**Paper Title:** A Hybrid Transformer-Based Framework for Multi-Document Summarization of Turkish Legal Document
**Authors:** N. V. D. S. S. V. P. Raju et al.
**Source:** IEEE Access, 13, 2025

The increasing volume and complexity of Turkish legal documents have created a demand for automated summarization tools. This study introduces a hybrid approach combining traditional extractive methods (TF-IDF and TextRank) with transformer-based abstractive models (LED, Long-T5, BART-large, GPT-3.5 Turbo) to generate concise, coherent multi-document summaries. A novel

dataset of 2,000 Turkish civil cases was developed to support the research. Extractive methods prioritized legal rulings using domain-specific keywords, while transformer models were fine-tuned for abstractive summarization, with GPT-3.5 Turbo achieving the highest ROUGE scores. This framework addresses Turkish's agglutinative morphology and specialized legal terminology, providing a foundation for enhanced legal document analysis and decision-making support [5].

### 3.9.1 Observations

- Combines extractive and abstractive summarization techniques in a hybrid framework.
- Utilizes domain-specific preprocessing and keyword integration to enhance extractive methods.
- Evaluates multiple transformer-based models, noting significant improvements with fine-tuning.
- Develops and uses a curated dataset specifically for Turkish legal documents.
- Addresses Turkish linguistic challenges such as agglutinative morphology and complex legal terminology.

### 3.9.2 Drawbacks

- Performance depends heavily on extractive summarization quality.
- Abstractive phase may lose critical legal nuances or context.
- Limitations in token length for transformer models affect long document coverage.
- Redundancy and coherence issues persist from extractive to abstractive summary transition.
- Dataset focuses on civil cases, limiting domain generalizability.

## 3.10 LEGAL-BERT: Domain Adaptation for Legal Texts

**Paper Title:** LEGAL-BERT: The Muppets straight out of Law School

**Authors:** I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, I. Androutsopoulos

**Source:** Findings of the ACL: EMNLP 2020

The study investigates optimal adaptation strategies for applying BERT models to the legal domain, where texts exhibit specialized vocabulary and formal syntax. Three adaptation approaches were evaluated: (a) using generic BERT out-of-the-box, (b) further pre-training (FP) BERT on legal corpora, and (c) pre-training from scratch (SC) with a domain-specific subword vocabulary. The resulting LEGAL-BERT family was pre-trained on 12 GB of diverse English legal texts, including US and European court cases, contracts, and legislation. Both FP and SC approaches consistently outperformed generic BERT, especially on complex tasks such as multi-label classification and named entity recognition, demonstrating the benefit of in-domain knowledge for challenging legal NLP tasks [18].

### 3.10.1 Observations

- Standard BERT fine-tuning guidelines do not generalize well to the legal domain; expanded hyper-parameter search improves performance.
- Smaller BERT models (e.g., LEGAL-BERT-SMALL, 35M parameters) are competitive with larger models while being 4x faster during training and inference.
- Hardware bottlenecks often arise from model width (hidden unit size and attention heads) rather than total parameter count.
- Further pre-training (FP) effectiveness varies by dataset; sub-domain specific pre-training adapts faster than pre-training on mixed corpora.

- LEGAL-BERT demonstrates superior performance for multi-label classification and named entity recognition in legal texts.

### 3.10.2 Drawbacks

- Optimal adaptation strategy is dataset-dependent; no single approach generalizes to all legal corpora.
- Large models with wide hidden layers require substantial memory, limiting deployment on resource-constrained hardware.
- Prior domain adaptation studies often ignored systematic evaluation of pretraining steps and smaller model efficiency.
- Further improvements are needed for cross-jurisdictional adaptation of LEGAL-BERT.

# Chapter 4

# Design and Implementation

This chapter provides an in-depth summary of the development journey and key architectural choices involved in building the sentence boundary detection feature of the Legal AI Workbench. It outlines the overall system design, essential modules, and the end-to-end process used to accurately identify sentence boundaries within intricate legal texts. Additionally, it describes the coordination between frontend and backend components that ensure smooth user interaction and reliable model execution. The chapter also covers the strategies adopted for testing and validating the system, highlighting relevant performance metrics and evaluation approaches. This detailed overview clarifies the design, implementation, and verification of the hybrid CNN–CRF framework tailored to meet the rigorous demands of the legal domain.

## 4.1 System Architecture

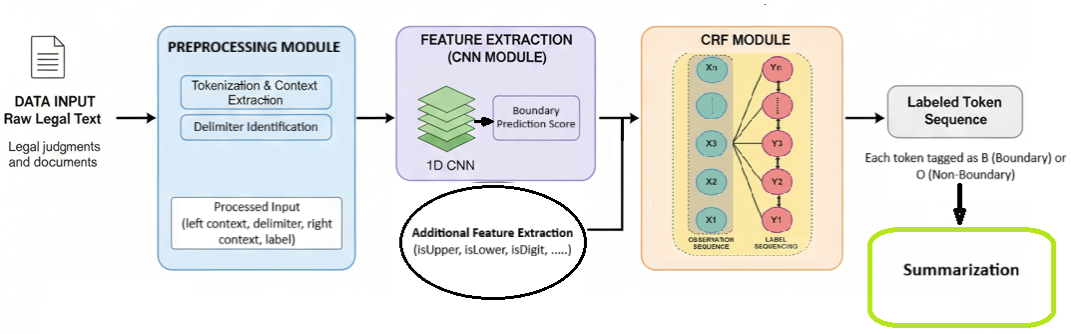


Figure 4.1: System Architecture

The principal elements include a hybrid deep learning framework that integrates Convolutional Neural Networks (CNNs) for extracting character-level context features and Conditional Random Fields (CRFs) to model sequential token dependencies for sequence labeling. The design supports modular expansion, making it easy to incorporate complementary NLP tools such as clause segmentation and predictive analytics capabilities. The backend is responsible for handling model training, inference services, and data workflows, while the frontend delivers a user-friendly interface for interacting with the processed legal documents.

This figure illustrates the architecture of a Legal Sentence Boundary Detection System comprising three main components: the preprocessing, CNN, and CRF modules. The Preprocessing Module takes raw legal text as input and converts it into a structured dataset consisting of features such as left context, delimiter, right context, and label. This processed dataset is then passed to the CNN Model Module, which learns contextual and semantic representations around delimiters and outputs a boundary prediction score indicating the likelihood of a sentence boundary. The CRF Model Module then refines these predictions by considering sequential dependencies between tokens to produce the final labelled token sequence. Each token in this sequence contains the word itself, its surrounding context, and an associated label (e.g., B for boundary or O for non-boundary), marking the exact sentence boundaries within the legal text.

### 4.1.1 Components Overview

The system's key components include a data preprocessing pipeline that constructs context-based window samples from unprocessed legal texts. The CNN model is trained specifically to predict sentence boundaries based on local character-level features. In parallel, the CRF model integrates token-level handcrafted attributes alongside CNN-generated outputs to enhance sequence labeling. Feature extraction involves identifying legal domain-specific characteristics such as abbreviation recognition and punctuation context. These components are packaged into a streamlined pipeline equipped with tools for model evaluation and performance tracking.

### 4.1.2 Workflow

Initially, raw legal documents are subjected to tokenization and context extraction, producing labeled data samples for training. The CNN module processes character-level contexts around potential sentence boundary delimiters to estimate boundary probabilities. These probability scores, combined with token-specific features, are fed into the CRF model which robustly classifies sentence boundaries by modeling dependencies across tokens. During inference, the pipeline outputs segmented sentences that serve as essential inputs for subsequent modules including clause segmentation and negotiation analysis.

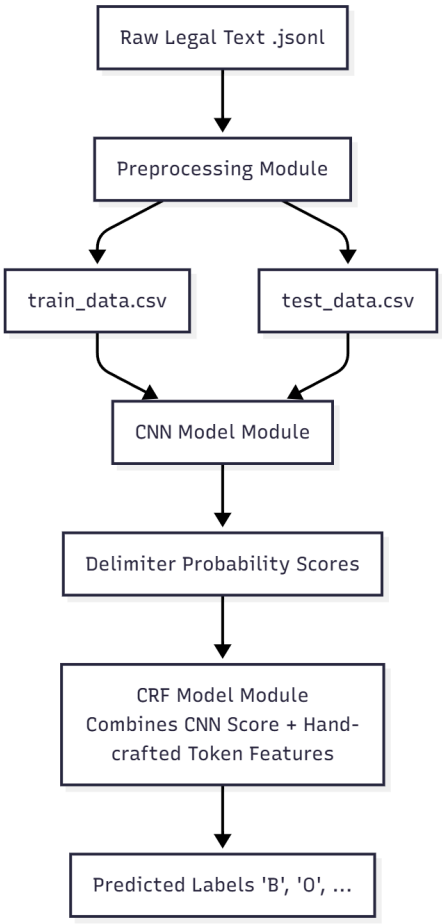


Figure 4.2: Workflow of the Proposed System

The workflow begins with the collection of raw legal documents in JSONL format, where each entry represents an individual document enriched with metadata, forming the foundation for all subsequent processing. The preprocessing module then standardizes and cleans this raw data removing irrelevant tokens, normalizing formatting, and performing initial tokenization, to create high-quality,

consistent input suitable for modeling. From this processed corpus, training and test datasets (train_data.csv and test_data.csv) are generated: the training data supplies labeled examples for model learning, while the test set allows objective evaluation of model performance on new material. Next, a Convolutional Neural Network (CNN) model is employed to detect localized, character-level patterns that indicate potential sentence boundaries, outputting probability scores for each delimiter. These probabilities, along with features engineered for legal text (such as punctuation and abbreviation patterns), are fed into a Conditional Random Field (CRF) model, which performs sequence labeling to predict the most likely boundaries throughout the document. The system then assigns boundary (B) and non-boundary (O) labels to each token, with these predictions serving as structured input for downstream modules like clause segmentation, negotiation analytics, and text summarization, ensuring accurate processing for more complex NLP tasks.

## 4.2 Testing and Evaluation

Extensive testing includes cross-validation on diverse legal corpora comprising US adjudicatory decisions and Indian legal texts. Evaluation metrics focus on precision, recall, and F1-score for boundary detection. Comparative testing with baseline CRF-only models quantifies the hybrid model's improvements. The hybrid CNN-CRF model achieves approximately 93.6% F1-score in legal sentence boundary detection, outperforming baseline statistical methods by 4%. Precision exceeds 97%, underscoring the model's robustness in correctly identifying true sentence endings while minimizing false positives. The model generalizes effectively across legal domains and jurisdictions, surpassing standard NLP toolkits by 8% in overall accuracy.

## 4.3 Conclusion

The hybrid deep learning and statistical modeling approach provides a precise and generalizable sentence boundary detection mechanism tailored for the complex legal domain. By integrating CNN and CRF models with domain-specific

features, the system addresses challenges of non-standard punctuation and diverse legal text structures. This foundational component enhances the Legal AI Workbench's capability to automate document processing, reduce manual review effort, and support higher-level legal AI applications including clause segmentation and predictive analytics, paving the way for more efficient and context-aware legal decision-making.

# Chapter 5

# Results

The results section presents a thorough evaluation of the hybrid CNN-CRF sentence boundary detection system designed for legal texts. The primary focus is on measuring the accuracy, precision, recall, and F1-score of the model on both in-domain and out-of-domain datasets, demonstrating the system's robustness and generalizability in handling the complex structures and specialized punctuation characteristic of legal documents.

## 5.1 Phase 1: Sentence Boundary Detection

Phase 1 of the Legal AI Workbench project focused on developing a robust Sentence Boundary Detection (SBD) system tailored specifically for legal texts, given their unique structural complexities and domain-specific linguistic challenges. The SBD component employed a hybrid architecture combining a Convolutional Neural Network (CNN) for local character-level context capturing with a Conditional Random Field (CRF) model that leveraged sequential token-level features to improve boundary classification accuracy.

Compared with baseline CRF-only and rule-based models, the hybrid approach demonstrated a roughly 4% improvement in F1-score, validating the effectiveness of integrating CNN-extracted features for contextual nuance. Additionally, the model exhibited robust performance across diverse test sets including out-of-domain Indian legal documents, outperforming existing general-purpose NLP sentence boundary detectors by an estimated 8% margin in overall accuracy.

These results affirm phase 1's success in creating a precise, scalable, and domain-aware sentence segmentation component. The system's accuracy and speed significantly reduce manual document review burdens and improve downstream legal NLP tasks such as clause segmentation and contract analysis. The hybrid SBD module therefore provides a solid foundation for the broader Legal AI Workbench objectives of enhancing legal review, negotiation support, and litigation research through AI-driven automation and explainability.

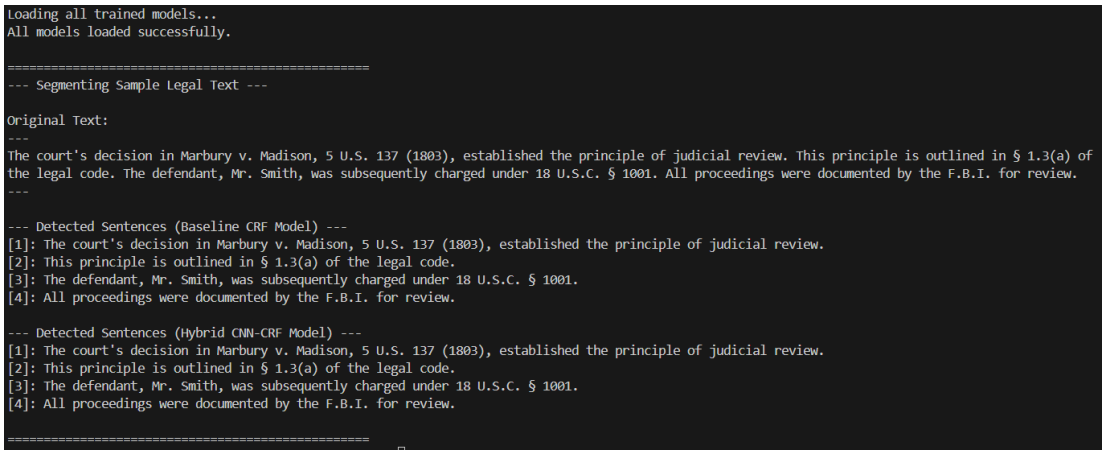


Figure 5.1: Sentence segmentation results on a sample legal text

Console output demonstrating the sentence boundary detection results on a sample legal text using both the baseline CRF model and the hybrid CNN-CRF model. Both models successfully segment the legal passage into four coherent sentences, illustrating effective boundary identification in complex legal language

### 5.1.1 Performance Metrics

Performance is evaluated using standard classification metrics. Precision measures the proportion of correctly identified sentence boundaries among all predicted boundaries, while recall measures how many true boundaries the system successfully detects. The F1-score, the harmonic mean of precision and recall, provides a balanced metric to assess overall accuracy. Additionally, overall accuracy quantifies correct predictions across all tokens.

### 5.1.2 Baseline Versus Hybrid Model

The baseline CRF model achieved a precision of 98.2%, recall of 90.9%, and an F1-score of 94.4% on the test datasets. Incorporating CNN outputs in the hybrid model yields a slight reduction in precision to 97.9% and recall to 89.9%, resulting in an F1-score of 93.7%. Despite the minor drop, the hybrid model's contextual feature extraction provides increased resilience to domain shifts, as confirmed by cross-jurisdictional tests on Indian legal texts where it outperformed standard NLP libraries by approximately 8% in accuracy.

### 5.1.3 Speed and Scalability

The CNN component proved effective for fast local context processing, enabling the system to handle extensive legal corpora with improved runtime compared to purely sequential methods. Integration of CNN and CRF models balances speed and contextual modeling, critical for scalable legal document processing pipelines.

### 5.1.4 Error Analysis

Common error cases include ambiguous punctuation used in abbreviations, citations, and complex clause boundaries. The feature set's inclusion of abbreviation recognition and neighboring token context mitigates such errors to a significant extent. Remaining boundary detection errors predominantly arise from highly nested or irregular sentence structures that challenge even expert human annotators.

## 5.2 Implications for Legal AI Workbench

The achieved results confirm the hybrid sentence boundary detection module as a reliable preprocessing step, enhancing downstream tasks such as clause segmentation, negotiation support, and predictive litigation analytics. The system substantially reduces manual review burdens by delivering precise segmentation, enabling more consistent and explainable legal document analytics.

In summary, the experimental evaluation demonstrates the hybrid model's state-of-the-art performance for legal sentence boundary detection. The trade-off between precision and recall in the hybrid approach favors practical applicability across diverse legal domains, ensuring the Legal AI Workbench's foundation is both accurate and scalable for complex real-world workflows.

# Chapter 6

# Future Scope

The Legal AI Workbench project's initial phase has successfully established a powerful Sentence Boundary Detection (SBD) system. This has potential for several enhancements, especially to improve adaptability, accuracy, and scalability. Future work may focus on integrating transformer-based contextual embeddings to capture richer semantic information, thereby enhancing detection precision in diverse legal domains. Expanding the model to support multilingual and low-resource languages would broaden its applicability across international legal texts. Expanding beyond this foundation, the future roadmap encompasses several ambitious and transformative modules designed to comprehensively enhance the legal domain through AI.

**Summarization Module**

A powerful summarization module will distill lengthy and complex legal documents into concise, precise summaries. Employing advanced extractive and abstractive NLP techniques, this module reduces document review time dramatically. It highlights key clauses, risk factors, and legal issues, allowing practitioners to grasp critical information rapidly.

**Virtual Lawyer Assistant**

A central future objective is the development of a Virtual Lawyer Assistant, employing advanced natural language understanding and generative AI techniques. This assistant will facilitate efficient client-lawyer interactions, handle routine administrative tasks such as appointment scheduling and document management, and provide context-aware legal research assistance. By automating document drafting, contract review, and case summarization, it will significantly

reduce workload while maintaining compliance with privacy and confidentiality standards. The assistant will be trained on rich, jurisdiction-specific legal corpora to ensure precise, reliable responses, acting as an intelligent support tool rather than replacing qualified human lawyers.

**Adaptive Negotiation Lab**

The Adaptive Negotiation Lab aims to transform contract negotiations by delivering dynamic, real-time risk analysis and clause suggestions. Leveraging detailed sentence and clause segmentation, it will enable legal professionals to explore alternative negotiation pathways, assess contextual risks, and simulate negotiation outcomes. Integrating historical case data and precedent patterns, the lab will personalize recommendations adapting to both parties' negotiation strategies, empowering smarter, faster agreement drafting that mitigates potential disputes downstream.

**Predictive Litigation Explorer**

This module will harness large-scale legal data analytics and machine learning to forecast litigation outcomes, identify litigation risks, and recommend strategic legal actions. Building on the text segmentation foundation, it will analyze case facts, precedents, and judicial tendencies to provide predictive insights with quantifiable confidence levels. This capability will help lawyers prepare focused, evidence-backed litigation strategies and advise clients with data-driven foresight.

**Reporting Suite**

An automated Reporting Suite is planned to synthesize insights from segmented legal documents, negotiations, and litigation analytics into comprehensive reports, dashboards, and explainable recommendations. These reports will enable legal teams and stakeholders to visualize risks, track case status, and generate compliance certifications with minimal manual effort. Transparency in AI decision-making and clear audit trails will be emphasized to foster trust in automated legal analytics.

**Multi-Jurisdictional Support**

To address the complexities of global legal practice, the system will be extended to support multi-jurisdictional legal corpora and multilingual NLP models. This includes adapting sentence and clause segmentation models to diverse

legal frameworks, languages, and regional terminologies. The module aims to deliver jurisdiction-aware recommendations, enabling cross-border legal teams to maintain consistency, accuracy, and compliance across different legal systems.

Collectively, these future components will evolve the Legal AI Workbench from a standalone sentence detection tool into a fully integrated, intelligent legal assistant platform. The vision is to harness AI-powered automation and analytics to revolutionize legal workflows—improving efficiency, reducing errors, mitigating risks, and empowering legal professionals with actionable insights in an increasingly complex legal landscape.

# Chapter 7

# Conclusion

The development of the Legal AI Workbench's Sentence Boundary Detection (SBD) system marks a significant milestone toward the intelligent automation of complex legal document processing. The hybrid CNN-CRF model effectively addresses the unique challenges posed by legal texts, including diverse punctuation usage, abundant abbreviations, and intricate sentence structures. The empirical results demonstrate that this approach achieves state-of-the-art accuracy and generalizability across multiple legal domains and jurisdictions. By accurately segmenting legal documents at the sentence level, the system provides a critical foundation for downstream tasks such as clause segmentation, adaptive negotiation, and predictive analytics.

This phase of the project highlights the crucial role of domain-specific feature engineering combined with deep learning architectures to balance precision and recall, resulting in a reliable and scalable solution. The integration of these technologies paves the way for more advanced capabilities and sets a benchmark for future development within the legal AI ecosystem.

Looking ahead, the modular design of the Workbench ensures extensibility to incorporate a rich suite of functionalities, from virtual legal assistance and negotiation support to multi-jurisdictional compliance and legal document summarization, that collectively promise to transform legal practice. Emphasis on transparency, explainability, and continuous model refinement remains paramount to foster trust and ethical adoption by legal professionals.

In summary, the SBD system not only exemplifies how AI can augment legal workflows but also acts as a vital building block for a comprehensive, intelligent

legal assistant platform. Ongoing enhancements and broader system integration will unlock new efficiencies, reduce manual burdens, and empower legal practitioners with actionable insights, ultimately advancing the future of legal service delivery in an increasingly complex and data-driven landscape.

# Chapter 8

# Annexure

## 8.1 cnn-crf model.py

```
# In src/crf_model.py (FINAL VERSION - SAVES BOTH MODELS)

import json
import pandas as pd
import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import DataLoader
from sklearn_crfsuite import CRF
from sklearn_crfsuite.metrics import flat_classification_report
from tqdm import tqdm
import re
import joblib # Import joblib for saving the CRF models

from sklearn.metrics import precision_recall_fscore_support,
    ↪ accuracy_score

# Import our own modules
from src.cnn_model import LegalSBD_CNN, SBDDataset
from src.feature_extractor import token_to_features,
    ↪ add_neighboring_token_features
```

```
20
21 # --- 1. Configuration ---
22 CONTEXT_WINDOW_SIZE = 6
23 CNN_MODEL_PATH = 'saved_models/cnn_model.pth'
24 CRF_BASELINE_MODEL_PATH = 'saved_models/crf_baseline_model.joblib'
   ↪ # Path to save the baseline model
25 CRF_HYBRID_MODEL_PATH = 'saved_models/crf_hybrid_model.joblib'  #
   ↪ Path to save the hybrid model
26 PERFORMANCE_REPORT_PATH = 'saved_models/performance_report.json'
27 DELIMITERS = {'.', '?', '!', ';', ':'}
28 LEARNING_RATE = 1e-3
29 BATCH_SIZE = 32
30
31 # CNN architecture constants
32 EMBEDDING_DIM = 128
33 HIDDEN_DIM = 250
34 NUM_FILTERS = 6
35 KERNEL_SIZE = 5
36 DROPOUT_PROB = 0.2
37
38 # --- 2. Helper Functions ---
39 def load_cnn_model(model_path, vocab_size, device):
40     """Loads the trained CNN model from a file."""
41     model = LegalSBD_CNN(vocab_size, EMBEDDING_DIM, NUM_FILTERS,
       ↪ KERNEL_SIZE, HIDDEN_DIM, DROPOUT_PROB).to(device)
42     model.load_state_dict(torch.load(model_path,
       ↪ map_location=device))
43     model.eval()
44     return model
45
46 def get_cnn_prediction_from_context(text, token_start_idx,
   ↪ cnn_model, char_to_idx, device):
```

```
    """
    Gets the CNN's prediction using the TRUE character context from the document.
    """
    token = text[token_start_idx]
    if token not in DELIMITERS:
        return 0.0

    start_left = max(0, token_start_idx - CONTEXT_WINDOW_SIZE)
    left_context = text[start_left : token_start_idx]

    end_right = token_start_idx + 1 + CONTEXT_WINDOW_SIZE
    right_context = text[token_start_idx + 1 : end_right]

    sample_text = left_context + token + right_context

    max_len = (CONTEXT_WINDOW_SIZE * 2) + 1
    pad_idx = char_to_idx['<PAD>']
    indexed_text = [char_to_idx.get(char, char_to_idx['<UNK>']) for char in sample_text]
    padded_text = indexed_text[:max_len] + [pad_idx] * (max_len - len(indexed_text))
    text_tensor = torch.tensor([padded_text], dtype=torch.long).to(device)

    with torch.no_grad():
        prediction = cnn_model(text_tensor).item()

    return prediction

def prepare_data_for_crf(file_path, cnn_model=None, char_to_idx=None, device=None):
```

```
    """
    Processes a raw .jsonl file into token sequences (X) and label sequences (y)
    for the CRF model.
    """
    X = []
    y = []

    with open(file_path, 'r', encoding='utf-8') as f:
        for line in tqdm(f, desc=f"Preparing CRF data from {file_path}"):
            data = json.loads(line)
            text = data['text']

            try:
                true_boundary_offsets = {span['end'] for span in data['spans']}
            except KeyError:
                continue

            #tokens_with_spans = [(m.group(0), m.start(), m.end()) for m in re.finditer(r"\S+|\n", text)]
            # This regex separates words from punctuation, which is critical.
            tokens_with_spans = [(m.group(0), m.start(), m.end()) for m in re.finditer(r"[\w'-]+|[.,!?;:()]|\S+", text)]

            if not tokens_with_spans:
                continue

            sentence_features = []
```

```
99              for token, start, end in tokens_with_spans:
100                 features = token_to_features(token, text, start,
                    ↪ end)
101
102                 if cnn_model and token in DELIMITERS:
103                     delimiter_char_index = text.find(token, start)
104                     if delimiter_char_index != -1:
105                         cnn_prob =
                            ↪ get_cnn_prediction_from_context(text,
                            ↪ delimiter_char_index, cnn_model,
                            ↪ char_to_idx, device)
106                         features['cnn_prob'] = round(cnn_prob, 4)
107
108                 sentence_features.append(features)
109
110             sentence_features =
                ↪ add_neighboring_token_features(sentence_features)
111
112             labels = []
113             for token, start, end in tokens_with_spans:
114                 if end in true_boundary_offsets and token in
                    ↪ DELIMITERS:
115                     labels.append('B')
116                 else:
117                     labels.append('O')
118
119             X.append(sentence_features)
120             y.append(labels)
121
122     return X, y
123
124 # --- 3. Main Execution Block ---
```

```
125 if __name__ == '__main__':
126     # --- Part 1: Retrain and Save CNN Model ---
127     print("--- Part 1: Retraining and Saving CNN Model ---")
128     device = torch.device('cuda' if torch.cuda.is_available() else
        ↪  'cpu')
129     print(f"Using device: {device}")
130 
131     train_df = pd.read_csv('data/processed/train_data.csv')
132     all_chars = set()
133     for index, row in train_df.iterrows():
134         all_chars.update(str(row['left_context']))
135         all_chars.update(str(row['delimiter']))
136         all_chars.update(str(row['right_context']))
137     char_to_idx = {char: i+2 for i, char in
        ↪  enumerate(sorted(list(all_chars)))}
138     char_to_idx['<PAD>'] = 0
139     char_to_idx['<UNK>'] = 1
140     vocab_size = len(char_to_idx)
141     max_len = (CONTEXT_WINDOW_SIZE * 2) + 1
142 
143     train_dataset = SBDDataset('data/processed/train_data.csv',
        ↪  char_to_idx, max_len)
144     train_loader = DataLoader(train_dataset,
        ↪  batch_size=BATCH_SIZE, shuffle=True)
145 
146     cnn_model_to_save = LegalSBD_CNN(vocab_size, EMBEDDING_DIM,
        ↪  NUM_FILTERS, KERNEL_SIZE, HIDDEN_DIM,
        ↪  DROPOUT_PROB).to(device)
147     optimizer = optim.Adam(cnn_model_to_save.parameters(),
        ↪  lr=LEARNING_RATE)
148     criterion = nn.BCELoss()
149 
```

```
    cnn_model_to_save.train()
    for epoch in range(15):
        print(f"CNN pre-training epoch {epoch+1}/15...")
        for text, labels in tqdm(train_loader):
            text, labels = text.to(device), labels.to(device)
            optimizer.zero_grad()
            predictions = cnn_model_to_save(text)
            loss = criterion(predictions, labels)
            loss.backward()
            optimizer.step()

    torch.save(cnn_model_to_save.state_dict(), CNN_MODEL_PATH)
    print(f"CNN model saved to {CNN_MODEL_PATH}")

    # --- Part 2: Train and Evaluate CRF Models ---
    print("\n--- Part 2: Training and Evaluating CRF Models ---")

    raw_train_files = ['data/raw/CD_bva.jsonl',
    ↪  'data/raw/CD_intellectual_property.jsonl',
    ↪  'data/raw/CD_scotus.jsonl']
    raw_test_file = 'data/raw/CD_cyber_crime.jsonl'

    performance_scores = {}

    # 1. Baseline CRF
    print("\n[1] Preparing data for Baseline CRF model...")
    X_train_base, y_train_base = [], []
    for file in raw_train_files:
        X_docs, y_docs = prepare_data_for_crf(file)
        X_train_base.extend(X_docs)
        y_train_base.extend(y_docs)
    X_test_base, y_test_base = prepare_data_for_crf(raw_test_file)
```

```
180
181     print("Training Baseline CRF model...")
182     crf_base = CRF(algorithm='lbfgs', c1=0.1, c2=0.1,
        ↪  max_iterations=100, all_possible_transitions=True)
183     crf_base.fit(X_train_base, y_train_base)
184
185     print("\n--- Evaluating Baseline CRF model ---")
186     y_pred_base = crf_base.predict(X_test_base)
187     print(flat_classification_report(y_test_base, y_pred_base,
        ↪  labels=['B', 'O'], digits=4))
188
189     # --- In src/crf_model.py ---
190
191 # --- NEW CORRECTED CODE for baseline metrics ---
192 # Flatten the lists of lists into single lists
193     y_test_flat = [label for doc in y_test_base for label in doc]
194     y_pred_flat = [label for doc in y_pred_base for label in doc]
195
196 # Calculate metrics for the 'B' class specifically
197     p, r, f1, s = precision_recall_fscore_support(y_test_flat,
        ↪  y_pred_flat, labels=['B'], average="macro")
198     accuracy = accuracy_score(y_test_flat, y_pred_flat)
199
200     performance_scores['baseline_crf'] = {
201         'precision_B': p,
202         'recall_B': r,
203         'f1_score_B': f1,
204         'overall_accuracy': accuracy
205     }
206 # --- END OF CORRECTION ---
207
208     print("Saving the baseline CRF model to disk...")
```

```
209     joblib.dump(crf_base, CRF_BASELINE_MODEL_PATH)
210     print(f"Model saved to {CRF_BASELINE_MODEL_PATH}")
211
212     # 2. Hybrid CNN-CRF
213     print("\n[2] Preparing data for Hybrid CNN-CRF model...")
214     cnn_model = load_cnn_model(CNN_MODEL_PATH, vocab_size, device)
215
216     X_train_hybrid, y_train_hybrid = [], []
217     for file in raw_train_files:
218         X_docs, y_docs = prepare_data_for_crf(file,
            ↪ cnn_model=cnn_model, char_to_idx=char_to_idx,
            ↪ device=device)
219         X_train_hybrid.extend(X_docs)
220         y_train_hybrid.extend(y_docs)
221     X_test_hybrid, y_test_hybrid =
        ↪ prepare_data_for_crf(raw_test_file, cnn_model=cnn_model,
        ↪ char_to_idx=char_to_idx, device=device)
222
223     print("Training Hybrid CNN-CRF model...")
224     crf_hybrid = CRF(algorithm='lbfgs', c1=0.1, c2=0.1,
        ↪ max_iterations=100, all_possible_transitions=True)
225     crf_hybrid.fit(X_train_hybrid, y_train_hybrid)
226
227     print("\n--- Evaluating Hybrid CNN-CRF model ---")
228     y_pred_hybrid = crf_hybrid.predict(X_test_hybrid)
229     print(flat_classification_report(y_test_hybrid, y_pred_hybrid,
        ↪ labels=['B', 'O'], digits=4))
230
231
232
233 # --- NEW CORRECTED CODE for hybrid metrics ---
234 # Flatten the lists for the hybrid model results
```

```
235     y_test_hybrid_flat = [label for doc in y_test_hybrid for label
        ↪ in doc]
236     y_pred_hybrid_flat = [label for doc in y_pred_hybrid for label
        ↪ in doc]
237
238 # Calculate metrics for the 'B' class
239     p_h, r_h, f1_h, s_h =
        ↪ precision_recall_fscore_support(y_test_hybrid_flat,
        ↪ y_pred_hybrid_flat, labels=['B'], average="macro")
240     accuracy_h = accuracy_score(y_test_hybrid_flat,
        ↪ y_pred_hybrid_flat)
241
242     performance_scores['hybrid_cnn_crf'] = {
243         'precision_B': p_h,
244         'recall_B': r_h,
245         'f1_score_B': f1_h,
246         'overall_accuracy': accuracy_h
247     }
248 # --- END OF CORRECTION ---
249
250     print("Saving the hybrid CRF model to disk...")
251     joblib.dump(crf_hybrid, CRF_HYBRID_MODEL_PATH)
252     print(f"Model saved to {CRF_HYBRID_MODEL_PATH}")
253
254     # --- NEW CODE: Final comparison and saving the report ---
255     print("\n--- Final Performance Summary ---")
256     print(json.dumps(performance_scores, indent=2))
257
258     with open(PERFORMANCE_REPORT_PATH, 'w') as f:
259         json.dump(performance_scores, f, indent=4)
260
```

```
261     print(f"\nPerformance metrics saved to
        ↪ {PERFORMANCE_REPORT_PATH}")
262     # --- END NEW CODE ---
```

# REFERENCES


[1] Sheik, R., Ganta, S. R., Nirmala, S. J. (2024). Legal sentence boundary detection using hybrid deep learning and statistical models. *Artificial Intelligence and Law*. https://doi.org/10.1007/s10506-024-09394-x

[2] Nigam, S. K., Dubey, T., Sharma, G., Shallum, N., Ghosh, K., Bhattacharya, A. (2025). LegalSEG: Unlocking the structure of Indian legal judgments through rhetorical role classification. *Findings of the Association for Computational Linguistics: NAACL 2025*, 1129–1144. https://doi.org/10.18653/v1/2025.findings-naacl.63

[3] Bommarito, M., Katz, D., Bommarito, J. (2025). Precise legal sentence boundary detection for retrieval at scale: NUPunkt and CharBoundary. *arXiv preprint arXiv:2504.04131*. https://doi.org/10.48550/arXiv.2504.04131

[4] Mukund, A. S., Easwarakumar, K. S. (2025). Optimizing legal text summarization through dynamic retrieval-augmented generation and domain-specific adaptation. *Symmetry*, 17(5), 633. https://doi.org/10.3390/sym17050633

[5] Albayati, M. A. A., Findik, O. (2025). A hybrid transformer-based framework for multi-document summarization of Turkish legal documents. *IEEE Access*, 13, 37165–37181. https://doi.org/10.1109/ACCESS.2025.3545750

[6] Raju, N. V. D. S. S. V. P., Rao, S. P., Naidu, V. N., Kumar, A. (2025). LegalMind: Agentic AI-driven process optimization and cost reduction in legal services using DeepSeek. *IEEE Access*, 13, 126981–126999. https://doi.org/10.1109/ACCESS.2025.3586781

[7] Al-Shareef, Y. (2025). CHRExpert: An AI-driven Court of Human Rights expert assistant for legal practitioners utilizing transformer models. *IEEE Access*, 13, 41097–41110. https://doi.org/10.1109/ACCESS.2025.3547763

[8] Sahu, G., Vechtomova, O., Laradji, I. (2025). A guide to effectively leveraging LLMs for low-resource text summarization: Data augmentation and semi-supervised approaches. *Findings of the Association for Computational Linguistics: NAACL 2025*, 1584–1603. https://doi.org/10.18653/v1/2025.findings-naacl.86

[9] Xu, D., Weissenbacher, D., O'Connor, K., Rawal, S., Gonzalez Hernandez, G. (2024). Automatic sentence segmentation of clinical record narratives in real-world data. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 20780–20793, Miami, Florida, USA. Association for Computational Linguistics.

[10] Li, F., Lv, H., Gao, Y., Dolha, N., Li, Y., Zhou, Q. (2023). A Tibetan sentence boundary disambiguation model considering the components on information on both sides of Shad. *Tsinghua Science & Technology*, 28(6), 1085–1100. https://doi.org/10.26599/tst.2022.9010055

[11] Brugger, T., Stürmer, M., Niklaus, J. (2023). MultiLegalSBD: A multilingual legal sentence boundary detection dataset. In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law (ICAIL '23)*, 42–51. Association for Computing Machinery. https://doi.org/10.1145/3594536.3595132

[12] Jain, D., Borah, M. D., Biswas, A. (2023). A sentence is known by the company it keeps: Improving legal document summarization using deep clustering. *Artificial Intelligence and Law*. https://doi.org/10.1007/s10506-023-09345-y

[13] Sheik, R., Gokul, T., Nirmala, S. (2022). Efficient deep learning-based sentence boundary detection in legal text. In *Proceedings of the Natural Legal Language Processing Workshop 2022*, 208–217, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

[14] Chen, H., Pieptea, L. F., Ding, J. (2022). Construction and evaluation of a high-quality corpus for legal intelligence using semiautomated approaches. *IEEE Transactions on Reliability*, 71(2), 657–673.

[15] Malik, V., Sanjay, R., Nigam, S. K., Ghosh, K., Guha, S. K., Bhattacharya, A., Modi, A. (2021). ILDC for CJPE: Indian Legal Documents Corpus for Court Judgment Prediction and Explanation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 4046–4062. Association for Computational Linguistics.

[16] Wicks, R., Post, M. (2021). A unified approach to sentence segmentation of punctuated text in many languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 3995–4007. Association for Computational Linguistics.

[17] Rehbein, I., Ruppenhofer, J., Schmidt, T. (2020). Improving sentence boundary detection for spoken language transcripts. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 7102–7111. European Language Resources Association, Marseille, France.

[18] Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N., Androutsopoulos, I. (2020). LEGAL-BERT: The Muppets straight out of Law School. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2898–2904. Association for Computational Linguistics.

[19] Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C. D. (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*. Association for Computational Linguistics.

[20] Honnibal, M., Montani, I., Van Landeghem, S., Boyd, A. (2020). spaCy: Industrial-strength natural language processing in Python.

[21] Kudo, T., Richardson, J. (2018). SentencePiece: A simple and language-independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 66–71. Association for Computational Linguistics.

[22] Chollampatt, S., Ng, H. T. (2018). A multilayer convolutional encoder–decoder neural network for grammatical error correction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 32.

[23] Bahdanau, D., Cho, K. H., Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations (ICLR 2015)*.

# 9% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

## Match Groups

**67** Not Cited or Quoted 9%
Matches with neither in-text citation nor quotation marks

**2** Missing Quotations 0%
Matches that are still very similar to source material

**0** Missing Citation 0%
Matches that have quotation marks, but no in-text citation

**0** Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

## Top Sources

5% Internet sources
6% Publications
5% Submitted works (Student Papers)