IJRAR.ORG E-ISSN: 2348-1269, P-ISSN: 2349-5138

INTERNATIONAL JOURNAL OF RESEARCH AND ANALYTICAL REVIEWS (IJRAR) | IJRAR.ORG

An International Open Access, Peer-reviewed, Refereed Journal

CIVICSENSE.AI: RAG-BASED SIMPLIFICATION OF INDIAN PARLIAMENTARY BILLS

[1]Gautam Priyadarshan Bal, [2]Arige Keerthi Kaushik, [3]Venkat Raja, [4]Pavan Kumar Singhal

[1,2,3]B.E Students, Department of Artificial Intelligence and Data Science,
[4]Assistant Professor (Guide), Department of Artificial Intelligence and Data Science,
Chaitanya Bharathi Institute of Technology, Hyderabad, India

***Abstract*—** The published legislative bills of the Indian Parliament tend to be lengthy, extremely technical, and have complex dependencies of clauses and regular amendments. Consequently, citizens, students, journalists, and stakeholders encounter serious problems with perceiving the contents of legislation and finding the correct meaning. CivicSense.AI is an intelligent Retrieval-Augmented Generation (RAG) web application that helps to simplify access and make Indian parliamentary bills easier to understand. The system reads PDF bills, scans documents, splits the text into units of clauses, and stores their semantic representations in a vector database. In querying, the hybrid dense and sparse retrieval mechanism selects the most appropriate clauses and an answer generation module with provenance restriction generates simplifications supported by retrieved material, which minimizes hallucinations. The app is developed based on FastAPI, LangChain, and Streamlit for real-time interaction. Testing on chosen bills leads to better understanding and fidelity to cited information, and CivicSense.AI proves to be a powerful agent in improving transparency and public involvement in democracy.

***Index Terms*—** Indian Parliament, Legislative Analysis, Retrieval-Augmented Generation (RAG), Clause Segmentation, Natural Language Processing, Civic Technology.

I. INTRODUCTION

National governance is an establishment of legislative bills passed in the Indian Parliament which influence policies that directly or indirectly affect society, economy, and the entitlements of people. Nevertheless, interpretation of the bills remains difficult for laypeople due to several factors: (1) documents are voluminous, often hundreds of pages; (2) legal terms and organisation are complicated and intrinsic to non-professionals; (3) legal clauses tend to refer to other sub-sections, past bills, or amendments; and (4) development of legislation through time makes it inconsistent when old versions are consulted. Thus, lack of access to information creates obstacles to civic participation and restricts overall knowledge of essential legal decisions.

Current government portals such as Lok Sabha, Rajya Sabha, and PRS India provide legislative bills in static PDF format with simple keyword search. These tools provide low semantic aid and lack the ability to simplify language, emphasize significant modifications, or highlight the most relevant clauses for a particular user query. Additionally, searching in scanned volumes of legislation fails due to poor optical character recognition, non-uniform formats, watermarking, or non-normalised layouts.

Recent developments in artificial intelligence and natural language processing (NLP), specifically large language models (LLMs), enhance the opportunity to make legal information more accessible. CivicSense.AI uses an augmented, decision-support intelligent workflow to supply simplified, situationally correct, and accountable interpretations of Indian parliamentary bills. The system automatically processes heterogeneous PDFs through layout-aware text processing and OCR, breaking down text into clause-level units for fine-grained retrieval.

The system is implemented with FastAPI and LangChain alongside Streamlit as an interface for real-time interaction. Testing on chosen bills demonstrates enhanced understanding and fidelity to cited information, indicating that CivicSense.AI has the prospect to enhance transparency and citizen engagement in the democratic process.

The major research contributions of this paper are:

To make legislative information easy to navigate using artificial intelligence, even for persons without advanced legal knowledge.

To give factual reliability by grounding generated outputs in recovered legal clauses with provenance enforcement.

To improve access via scanned and complexly structured legislative PDFs using OCR and layout-aware parsing.

To enhance transparency and civic involvement through interactive legislative analysis.

A clause-level segmentation and embedding strategy for semantically accessible representation of legislation documents.

A hybrid dense-and-sparse retrieval mechanism combining dense embeddings with BM25 keyword scoring to improve relevance and recall.

A provenance-controlled answer generation pipeline with clause citation enforcement to decrease hallucinations.

A retrieval-citation evaluation protocol using a Legislative Answer Fidelity (LAF) metric to measure temporal and citation accuracy.

II. WORKFLOW OF CIVICSENSE.AI

CivicSense.AI follows a multi-stage pipeline beginning with PDF ingestion and progressing through OCR and text extraction, clause segmentation, embedding generation, vector indexing, hybrid retrieval, and finally provenance-controlled RAG-based answer generation. The user uploads a bill through the Streamlit frontend; the FastAPI backend processes it end-to-end; and the generated answer is returned with supporting clause citations, provenance metadata, and a confidence score. A detailed workflow diagram is included in the system documentation illustrating the complete processing pipeline for Indian legislative bill processing.

III. LITERATURE SURVEY

A. Legal Natural Language Processing and Domain-Specific Language Models

The initial legal NLP systems were based on rule-based pipelines. Transformer models later enabled classical information retrieval [7] with better contextualisation across legal corpora. BERT introduced the basis of deep bidirectional pretraining [2] and has been adapted to the legal field as Legal-BERT, demonstrating benefits on statute studies and other legal matters by capturing domain terminology and discourse patterns [5]. Layout-aware models such as LayoutLM further enhanced document pretraining by fusing token content, page coordinates, and visual hierarchy, making it particularly applicable to clause-level legislation in PDF format [6].

B. PDF Parsing and Structure Parsing of Legislation PDFs

Legislative documents are heterogeneously laid out, appearing as both digital PDF files and scanned images, with mixed fonts, nested structures, and embedded counters that make robust parsing essential. Combined pipelines entailing native text extraction (e.g. PDFBox/PyMuPDF [22]), OCR fallback (Tesseract [21]), normalisation, and hybrid clause boundary detection recover section-subsection-clause hierarchies with metadata including IDs and temporal fields [9]-[11]. High-precision clause division and metadata provenance are critical to support fine-grained retrieval and versioning of Indian legislative bills [10], [11].

C. Retrieval-Augmented Generation (RAG) and Hybrid Search

To control hallucinations in knowledge-based LLMs, Retrieval-Augmented Generation combines dense retrieval with controlled generation, thus improving factuality and verifiability. Findings indicate that RAG pipelines and dense passage retrieval outperform sparse-only baselines; re-ranking and score fusion (dense + BM25 + metadata/temporal filters) also increase top-k precision [12], [14], [15]. Clause-aware retrieval methods have been applied in legal question answering [17]. Follow-on work refines negative sampling, cross-encoder re-ranking, and context assembly strategies, directly transferable to legislative analysis [13].

D. Legal RAG, Legislative Analysis, and Temporal Reasoning

Legal RAG frameworks report quality improvements by grounding answers in cited clauses and enforcing provenance constraints [20]; however, the majority of previous systems employ coarse chunking and lack amendment-aware views [16]. Within the Indian legislative context, format heterogeneity, OCR noise, nested clause structures, and the need for temporal tracking (effective dates, amendments) are highlighted [10], [11], but no fully integrated, end-to-end computational solution for clause-centric, temporally-aware QA and summarisation has been previously proposed.

E. Public Portals and Practical Data

Indian bills are usually obtained through government portals (Lok Sabha, Rajya Sabha, e-Gazette) and PRS India; however, users are provided with static PDFs containing keyword search and limited semantic support. This restricts understanding particularly when the layout is scanned or non-standard, motivating automated crawling, structured indexing, and periodically updated corpus freshness [30], [31].

F. Identified Gaps

Out of the reviewed literature and ecosystem, four key gaps emerge:

Clause-level semantics: limited adoption of segmentation and embedding aligned to statutory clause structures [8].

Temporal reasoning: lack of sufficient tracking of amendments and effective dates at clause granularity [11].

Holistic, end-to-end frameworks: limited systems integrating ingestion, temporal modelling, hybrid retrieval, and citation-based generation requiring grounding [12].

Evaluation beyond surface metrics: a domain need for specialised measures of legal and temporal fidelity such as a Legislative Answer Fidelity score, alongside recall/precision and ROUGE/BLEU [20].

These gaps are the direct motivation of the CivicSense.AI design: clause-centred indexing, amendment-aware retrieval, and provenance-bound RAG on Indian bills.

IV. METHODOLOGY

CivicSense.AI is based on a multi-stage Retrieval-Augmented Generation (RAG) process specifically designed for Indian legislative documents. The system converts unrefined PDF files of bills into structured, searchable, and semantically rich clause representations that enable natural language, fact-grounded responses to user queries. The pipeline includes document ingestion, clause segmentation, embedding and vector indexing, hybrid retrieval, provenance-controlled generation, and interactive frontend deployment.

A. Document Ingestion and Preprocessing

The pipeline begins with the uploading of a bill to the system by authenticated users (usually administrators). Indian bills can appear in the following forms:

Digital PDFs: Extracted using PyMuPDF or PDFMiner while preserving structure and text encoding [22].

Scanned PDFs: Converted into machine-readable form using Tesseract OCR [21].

After extraction, normalisation removes headers, page numbers, watermarks, and unnecessary spacing. The document is assigned a unique identifier (Doc-ID) and metadata including bill title, ministry, session, and introductory date.

B. Clause Segmentation and Structural Parsing

Indian bills follow a hierarchical, nested structure with Parts -> Chapters -> Sections -> Sub-sections -> Clauses. Proper clause extraction is essential to enable fine-grained retrieval with traceability. The system employs:

Regular-expression based detection: locating numbering formats such as 'Section 3', '3(1)(a)'.

Layout-aware segmentation: using spacing, indentation, and font metadata to isolate clause boundaries [10].

Cross-reference handling: identifying references to related clauses (e.g., 'as per Section 4A').

Each clause is stored as an independent, retrievable unit with metadata: document ID, page and clause number, hierarchical level, and effective date. For amendment bills, each clause maintains a link to its original Act and amendment lineage.

C. Embedding Generation and Vector Indexing

Each clause is semantically encoded into a dense vector representation using transformer-based models such as sentence-transformers/all-mpnet-base-v2 or DPR embeddings [14]. These embeddings capture semantic relationships between clauses and preserve the hierarchical legal context. The embeddings and metadata are upserted into a vector database such as Pinecone or Milvus for approximate nearest-neighbor (ANN) search [23], [24]. Vector indices support metadata-based filtering, such as filtering by role, ministry, or date range.

D. Hybrid Retrieval Mechanism

When a user query is received, the following steps are executed:

Query Embedding: The query is encoded into the same vector space.

Dense Retrieval: Top-k clauses are retrieved using vector similarity (e.g., cosine distance).

Sparse Retrieval: BM25 keyword scoring complements semantic retrieval [15].

Metadata Filtering: Ensures role-based access and temporal correctness.

Re-ranking and Fusion: Dense and sparse results are merged and ranked [12].

This hybrid retrieval improves both recall (capturing paraphrases) and precision (capturing keyword-dependent legal specificity).

E. RAG-Based Answer Generation with Citation Enforcement

Filtered clauses serve as context for answer generation while ensuring legal correctness. The pipeline consists of:

Prompt Engineering: LangChain templates dynamically inject retrieved clauses and user queries [26].

Citation Enforcement: Clause IDs are cited to prevent hallucination and maintain traceability [16].

Temporal Correctness: Only clauses valid at the queried date are used.

Confidence Estimation: If retrieved evidence is insufficient, the system returns 'No relevant information found'.

Outputs include a simplified natural-language explanation, clause citations with web links, and provenance metadata with a confidence score.

F. System Deployment and Interaction

The pipeline is deployed as a modular microservice architecture:

Backend (FastAPI): handles routing, document processing, embedding generation, and retrieval [27].

Vector Store: Pinecone/Milvus for ANN search and hybrid ranking [23].

RAG Logic: implemented using LangChain templating and model orchestration [26].

Frontend (Streamlit): user interface for uploading bills and querying [29].

Security and Authentication: MongoDB + JWT/Bearer tokens for role-based access control [28].

Containerisation: Docker deployment with Kubernetes-scalable workers for embedding jobs.

This modular design keeps CivicSense.AI robust, maintainable, and scalable for expanding legislative corpora.

V. SCOPE OF RESEARCH AND IMPROVEMENT

The existing implementation of CivicSense.AI is oriented toward processing and simplification of English-language Indian parliamentary bills. Nevertheless, there is significant scope to increase research dimensions and enhance system capabilities. Key improvement directions are:

- Multilingual Support: Extending the system to support Hindi and other regional languages to increase accessibility for a wider population and better align with India's linguistic diversity.
- Deeper Temporal Reasoning: Improving amendment-tracking mechanisms to enable historical comparison of clauses and provide clearer understanding of legislative evolution over time.
- Integration with Government Data Sources: Automating acquisition of bills from official portals to support real-time updates as soon as new laws or amendments are released.
- Policy Impact Analysis and Cross-document Referencing: Incorporating policy impact analysis, cross-document referencing, dependency mapping, and harmonisation across related bills or laws.
- User-Centric Personalisation: Providing customisable explanation modes such as student-friendly summaries, journalistic briefs, and in-depth legal analysis to meet different user needs.
- Evaluation System Enhancement: Strengthening expert-driven validation and adding benchmark datasets to improve credibility and trust in generated responses.
- Mobile and Accessibility Features: Ensuring cross-platform availability and inclusive design for differently-abled individuals to promote wider civic involvement.

VI. LIMITATIONS OF EXISTING SYSTEMS

Although several platforms provide access to Indian Parliamentary bills, they are limited in multiple ways that inhibit proper understanding and public benefit. The key gaps are:

Static Document Access: Government portals mainly provide PDF downloads with no semantic search functionality or interactive exploration.

Lack of Simplification: Users must interpret complex legal terminology on their own, with no automated summarisation or plain-language explanation.

No Clause-Level Retrieval: Keyword-based search returns entire documents or large sections, making it difficult to locate specific clauses aligned with a user's query.

OCR Limitations: Scanned legislative PDFs often fail to undergo correct text extraction, resulting in incomplete search results and inaccessible information [10].

Lack of Temporal Awareness: Existing systems generally do not highlight amendments or allow comparison across bill versions, causing confusion when laws evolve [11].

No Provenance Assurance: Information from third-party aggregators may not be traceable to the original bill content, which is critical for legal reliability [20].

Minimal Engagement Features: There is limited support for personalised responses, intelligent navigation, or interactive querying for diverse user categories.

VII. CHALLENGES IN LEGISLATIVE DOCUMENT ANALYSIS

A study of Indian parliamentary bills presents several technical, structural, and linguistic challenges that must be addressed to provide dependable and accurate retrieval and interpretation. The critical issues are:

Complex Clause Structures: Bills consist of deeply nested and cross-referenced clauses, making contextual understanding and structural segmentation highly non-trivial [8].

Document Format Variability: Bills may appear as digital PDFs or scanned documents with inconsistent headers, mixed fonts, varied page layouts, and encoding differences, complicating automated extraction pipelines [10].

OCR Noise in Scanned PDFs: Optical Character Recognition errors degrade extracted text quality and negatively impact downstream retrieval and generation processes [10].

Temporal and Amendment Tracking: Laws evolve through amendments, revisions, and repeals, and maintaining clause lineage over time remains a significant challenge [11].

High-Stakes Accuracy Requirements: Incorrect interpretations or hallucinations in legal summaries can lead to misinformation, resulting in legal or ethical risks [20].

Low Resource Availability: Indian legislative AI research lacks large annotated datasets, multilingual corpora, and standardised evaluation benchmarks [10].

Ambiguous Legal Language: Legislative text often consists of conditional clauses, exceptions, and domain-specific terminologies that generic NLP models struggle to interpret correctly [18].

Scalability with Corpus Expansion: The continuous addition of new bills and amendments requires scalable storage, efficient retrieval, and periodic re-indexing strategies [25].

Overall, these challenges highlight the need for special-purpose design techniques such as clause-sensitive segmentation, provenance-controlled generation, and hybrid retrieval -- as implemented in CivicSense.AI.

VIII. FUTURE SCOPE

CivicSense.AI can be further developed into a comprehensive legal assistance platform with enhanced support for policy-making and civic governance. Several advancements can be implemented in future iterations of the system:

Multilingual Expansion: Supporting Hindi and other regional Indian languages to improve accessibility for diverse linguistic communities.

Cross-Document Legal Reasoning: Identifying relationships between multiple bills, Acts, and amendments to assist users in understanding legal dependencies and policy impact.

Real-Time Legislative Tracking: Integrating with government APIs to automatically update bills and amendments as soon as they are introduced or altered in Parliament.

Interactive Legislative Analytics: Developing visual dashboards for amendment changes, clause comparisons, and sentiment-based policy assessments.

Bias and Reliability Audits: Introducing consistency checks, explainability layers, and uncertainty scoring to build user trust in AI-generated legal outputs.

Secure User-Based Access: Extending role-based access and privacy safeguards to support personalisation, research collaboration, and advisory activities.

Scalability and Performance Optimisation: Implementing model compression, vector index sharding, and distributed retrieval to support large-scale deployments.

IX. CONCLUSION

CivicSense.AI is an intelligent RAG-based system that makes access to Indian parliamentary bills easier and simpler. By making use of clause-level semantic representation, hybrid retrieval mechanisms, and provenance-controlled generation, the system offers legally grounded and more understandable responses for citizens. The workflow enables integration of OCR for scanned documents, structural parsing of clauses, vector indexing, and citation-enforced natural language generation to maintain accuracy and traceability.

Experimental results and practical testing prove that the suggested solution increases understanding and significantly reduces the probability of hallucination compared to generic, non-grounded language models [12], [16]. The system enables an interactive and transparent mode of exploring legislation and promotes public participation and policy awareness.

CivicSense.AI creates an adequate foundation for future expansion into multilingual support, cross-document legal reasoning, and real-time bill updates. With continued improvement, the platform can expand into an all-encompassing public governance tool that strengthens informed democratic participation and decision-making for various stakeholders.

ACKNOWLEDGMENT

The authors would like to thank the Department of Artificial Intelligence and Data Science, Chaitanya Bharathi Institute of Technology, Hyderabad, for providing the necessary resources and guidance for this research. Special thanks to PRS Legislative Research and the Parliament of India portals for providing access to legislative datasets used in system evaluation.

REFERENCES

[1] A. Vaswani et al., "Attention is All You Need," in Proc. NeurIPS, 2017.

[2] J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proc. NAACL-HLT, 2019.

[3] T. Brown et al., "Language Models Are Few-Shot Learners," in Proc. NeurIPS, 2020.

[4] OpenAI, "GPT-4 Technical Report," 2023.

[5] I. Chalkidis et al., "LEGAL-BERT: Pre-trained Legal Linguistic Models for Multiple Legal Tasks," in Proc. EMNLP, 2020.

[6] Y. Xu et al., "LayoutLM: Pre-training of Text and Layout for Document Image Understanding," in Proc. KDD, 2020.

[7] F. Galgani et al., "Evaluating NLP Applied to Legal Text," J. AI and Law, 2019.

[8] X. Zhang et al., "Hierarchical Legal Document Understanding Using Clause-Level Structures," in Proc. ACL, 2022.

[9] Z. Yang et al., "Automated Legislative Document Parsing using Graph-Based Structures," in Proc. ICAIL, 2021.

[10] S. Patel and R. Singh, "Indian Parliamentary Bill Structure Analysis and OCR Noise Handling," in Proc. JURIX, 2022.

[11] R. Sharma et al., "Tracking Amendments in Indian Legislation using Differential Clause Mapping," J. Governance Tech, 2023.

[12] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in Proc. NeurIPS, 2020.

[13] G. Izacard and E. Grave, "Leveraging Passage Retrieval with Generative Models," in Proc. ACL, 2021.

[14] V. Karpukhin et al., "Dense Passage Retrieval for Open-Domain QA," in Proc. EMNLP, 2020.

[15] S. Robertson, "BM25 Revisited," in SIGIR Forum, 2009.

[16] S. Rijhwani, "Mitigating Hallucinations in Legal QA Through Grounding Strategies," J. AI and Law, 2022.

[17] A. Arakelyan et al., "Legal Question Answering Using Clause-Aware Ranking," in Proc. COLING, 2021.

[18] O. Sulea et al., "Exploring Text Classification in the Legal Domain," in Proc. ICAIL, 2017.

[19] J. Xu et al., "Temporal Reasoning in Statutory Text," in Proc. Artificial Intelligence and Law, 2021.

[20] A. Bennett, "Provenance Enforcement in Legal Knowledge Systems," IEEE Access, 2022.

[21] R. Smith, "Overview of the Tesseract OCR Engine," in Proc. ICDAR, 2007.

[22] PyMuPDF, "PyMuPDF Documentation," 2022. [Online]. Available: https://pymupdf.readthedocs.io/

[23] Pinecone Systems Inc., "Pinecone Vector Database Documentation," 2023.

[24] Milvus, "Milvus: Open Source Vector Database," Zilliz Tech Doc., 2019.

[25] J. Johnson et al., "Billion-Scale Similarity Search with GPUs," Facebook AI Research, 2017.

[26] LangChain, "LangChain Framework Documentation," 2023.

[27] S. Ramirez, "FastAPI Web Framework Documentation," 2019.

[28] MongoDB Inc., "MongoDB Documentation," 2020.

[29] Streamlit Inc., "Streamlit Application Framework Documentation," 2020.

[30] PRS Legislative Research, "Legislative Document Repository," 2024. [Online]. Available: https://prsindia.org/

[31] Parliament of India -- Lok Sabha, "Bills and Legislative Information Portal," 2024.