INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD)
© 2026 IJNRD | Volume 11, Issue 2, February 2026 | ISSN: 2456-4184 | IJNRD.ORG

AI-Based Legal Document Understanding and Case Query Assistant Using NLP and Transformer Models

R. Bhuvaneswari[1], S. Rupasri [2]

R. Bhuvaneswari[1], M.Sc., M.Phil., M.Tech., SET., Assistant Professor, Department of Computer Science and Applications D.K.M. College for Women (Autonomous)
S. Rupasri [2], PG Student, Department of computer Science and Applications D.K.M. College for Women (Autonomous), Vellore, Tamil Nadu, India.

***Abstract*—**The rapid growth of digital legal records has created an urgent need for intelligent systems capable of efficiently processing and interpreting complex legal documents. This paper presents an AI-Based Legal Document Understanding and Case Query Assistant that leverages Natural Language Processing (NLP), transformer-based language models, and Retrieval-Augmented Generation (RAG) to automate legal document analysis and enable intelligent case querying. The proposed system employs InLegalBERT, a domain-specific transformer model, for generating semantic embeddings of Indian Penal Code (IPC) sections and legal case texts. A FAISS vector store enables high-speed similarity search, while a hybrid retrieval mechanism combining structured JSON section lookup with semantic embedding search ensures accurate context retrieval. The Groq API with large language model inference generates concise, context-aware legal responses. Redis-based conversation management maintains multi-turn query history. Experimental results demonstrate that the system accurately identifies pertinent legal information, reduces research time significantly, and provides accessible legal insights to both professionals and non-experts. The system represents a practical integration of modern AI technologies into the legal domain, offering a scalable and extensible foundation for future legal analytics applications.

Index Terms*—*Legal AI, Natural Language Processing, Transformer Models, InLegalBERT, FAISS, Retrieval-Augmented Generation, Semantic Search, Legal Document Summarization, Indian Penal Code, Groq API, Redis, Python.

I. INTRODUCTION

The rapid growth of digital legal records has fundamentally transformed how legal information is stored, accessed, and analysed. Courts, law firms, and corporate legal departments generate vast volumes of contracts, case judgments, statutes, and regulatory documents every day. However, legal texts are often lengthy, complex, and written in highly technical language, making manual review time-consuming, costly, and prone to human oversight. This creates a strong and pressing need for intelligent systems that can efficiently process, understand, and retrieve relevant legal information on demand.

Traditional legal research relies on keyword-based search mechanisms provided by conventional legal databases. While these systems are useful for locating specific documents containing targeted phrases, they fundamentally lack the ability to interpret the semantic meaning behind legal language, which often involves complex terminologies, conditional clauses, nested citations, and contextual nuances that vary across jurisdictions and court levels. A keyword search for "breach of contract" may return thousands of unranked results without distinguishing between cases where the breach was upheld, dismissed, or settled, making the results unreliable for actionable legal research.

An AI-Based Legal Document Understanding and Case Query Assistant addresses these limitations by leveraging Artificial Intelligence (AI), Natural Language Processing (NLP), and Machine Learning (ML) to automate the analysis of legal documents and enable intelligent legal search. The system is designed to extract key information such as parties, legal provisions, obligations, judgments, and dates, while also supporting natural language queries for case law research and document summarization. By integrating transformer-based language models and retrieval techniques, the assistant can provide context-aware responses, summarize lengthy judgments, identify relevant precedents, and assist in legal reasoning.

The proposed system is built upon four complementary AI technologies. First, InLegalBERT, a transformer model specifically pretrained on Indian legal corpora, generates domain-adapted semantic embeddings that capture the specialized vocabulary and reasoning patterns of Indian law. Second, a FAISS (Facebook AI Similarity Search) vector store enables sub-second similarity retrieval across thousands of embedded legal passages. Third, Retrieval-Augmented Generation (RAG) combines the retrieved context with generative LLM inference to produce accurate and grounded legal responses. Fourth, Redis-based conversation management preserves multi-turn query history, enabling coherent follow-up questions within a legal research session.

The primary contributions of this work are: (1) a complete end-to-end legal AI pipeline integrating document preprocessing, semantic embedding, hybrid retrieval, and generative response synthesis; (2) application of InLegalBERT for Indian legal domain-specific embedding with FAISS indexing; (3) a hybrid retrieval strategy combining structured JSON section matching

with FAISS semantic search; (4) multi-turn conversation management using Redis for persistent query history; and (5) deployment as an interactive web-based assistant accessible to legal professionals, law students, and researchers.

II. LITERATURE SURVEY

A. Legal Question Answering Using Deep Learning and Semantic Search (Okumura et al., 2025)

Okumura, Matsumoto, and Yoshikawa presented a deep learning-based legal question answering system that encodes legal documents including statutes, contracts, and court judgments into semantic embeddings using transformer models. Queries are matched against these embeddings to provide context-aware responses. Experiments demonstrated accurate identification of pertinent case information and legal clauses, significantly improving legal research efficiency. The study emphasizes the role of semantic representations in bridging the gap between complex legal language and end-user accessibility, directly motivating the embedding-based retrieval design of the proposed system.

B. Automated Summarization of Legal Decisions Using Transformer Models (Gupta et al., 2025)

Gupta, Sharma, and Gupta focused on automatic summarization of legal documents through transformer-based models, applying both extractive and abstractive summarization to highlight key sections and legal reasoning from court decisions. Tests on a corpus of judgments showed that transformer models outperform traditional summarization methods in readability and relevance. The work demonstrates how AI can reduce manual effort and enhance accessibility of legal information for both professionals and learners, informing the summarization module architecture of the proposed assistant.

C. NLP for Legal Document Analysis (Branting et al., 2024)

Branting, Martinez, and Knoblock demonstrated the application of NLP for extracting structured information from legal texts using linguistic parsing, entity recognition, and semantic classification applied to case judgments and statutes. The system improved retrieval of legal information compared to keyword-based search by understanding contextual meaning. Challenges identified include handling complex legal syntax and domain-specific terms, which the proposed system addresses through the use of InLegalBERT, a model specifically trained on Indian legal corpora rather than general-domain text.

D. Context-Aware Query Interpretation (Udayakumar et al., 2025)

Udayakumar et al. introduced CAQI, a context-aware legal information retrieval framework combining user profiling, query reformulation, and ontology-driven semantic enrichment to enhance retrieval relevance. Legal ontologies resolve term ambiguity and capture hierarchical relationships, while hybrid ranking strategies incorporate semantic embeddings with lexical similarity measures. Evaluation with legal professionals showed significant gains in accuracy and relevance over traditional keyword and semantic baseline systems, validating the hybrid retrieval approach—combining JSON section lookup with FAISS semantic search—adopted in the proposed system.

E. Multi-Layered Embedding-Based Retrieval (Lima, 2024)

Lima proposed a multi-layered embedding retrieval method for legal and legislative texts, creating dense vector representations at multiple structural levels—articles, clauses, and chapters—to capture legal meaning and hierarchy. This work demonstrates that legal documents have a natural hierarchical structure that should be reflected in embedding design, informing the section-level embedding strategy used to index IPC provisions in the proposed FAISS vector store.

F. Transformer-Based Legal Information Retrieval (Kim et al., 2024)

Kim, Rabelo, Babiker, Rahman, and Goebel explored transformer-based models for legal information retrieval and entailment, demonstrating that deep transformer architectures outperform classical retrieval techniques in aligning legal queries with contextually relevant documents, improving both recall and entailment accuracy. This work validates the use of transformer embeddings as the primary retrieval mechanism in the proposed system and motivated the selection of FAISS for approximate nearest-neighbour search over the legal embedding index.

G. LSTM-Based Legal NER and Semantic Classification (Naik et al., 2024)

Naik, R. K., and Patel presented an LSTM-based approach to legal named entity recognition and semantic classification, identifying key elements within legal texts and assigning semantic labels to enhance semantic search effectiveness. While LSTM-based NER achieves reasonable accuracy, the comparison with transformer-based models highlights the superiority of attention mechanisms for capturing long-range dependencies in complex legal sentences. This finding reinforces the selection of InLegalBERT over LSTM-based alternatives for embedding generation in the proposed system.

III. EXISTING SYSTEM AND LIMITATIONS

Currently, legal research and document analysis rely predominantly on manual processes or simple keyword-based search mechanisms provided by traditional legal databases such as Manupatra, SCC Online, and India Kanoon. Lawyers, paralegals, and researchers often need to manually read through extensive case judgments, contract clauses, or regulatory texts to extract relevant information, which is both time-intensive and prone to human error. Existing platforms provide basic search functionalities allowing users to locate documents containing specific keywords, phrases, or legal citations, but fundamentally lack the ability to interpret semantic meaning.

These systems cannot answer natural language questions, provide concise summaries, or highlight key legal points within a document, leaving users to sift through pages of information to find relevant content. This limitation is particularly significant when dealing with large-scale legal datasets—the Supreme Court of India alone has issued over 30,000 judgments indexed

online—where keyword searches yield too many irrelevant documents or fail to capture subtle relationships between cases involving similar legal principles but different terminology. In educational settings, law students face challenges comprehending long judgments, as existing tools provide no interactive guidance or context-aware analysis.

Table I: Existing Systems vs. Proposed System

Feature	Existing Systems	Proposed System
Search Method	Keyword matching only	Hybrid semantic + JSON retrieval
Legal Understanding	None (syntactic only)	InLegalBERT embeddings
Query Interface	Boolean / phrase search	Natural language queries
Summarization	Not available	LLM-generated concise summaries
Domain Adaptation	General purpose	Indian legal corpus (IPC)
Conversation History	Not maintained	Redis multi-turn sessions
Response Generation	Document links only	RAG-grounded explanations

IV. PROPOSED SYSTEM

The proposed system is an AI-Based Legal Document Understanding and Case Query Assistant designed to simplify the handling and interpretation of legal texts through four tightly integrated stages: document preprocessing and embedding, FAISS vector store construction, hybrid retrieval, and generative response synthesis. Legal documents such as IPC sections, case judgments, statutes, and contracts are processed into structured semantic representations. Users interact with the system through natural language queries, receiving concise and context-aware legal responses grounded in retrieved legal passages rather than hallucinated information.

A. System Objectives

- Process and embed IPC sections and legal case documents into semantic vector representations using InLegalBERT for domain-specific legal understanding.
- Build and maintain a FAISS vector index enabling sub-second approximate nearest-neighbour search across the embedded legal corpus.
- Implement a hybrid retrieval mechanism combining explicit IPC section number JSON lookup with FAISS semantic similarity search for maximum recall.
- Generate accurate, concise, and contextually grounded legal responses using Groq API large language model inference with RAG prompting.
- Maintain multi-turn conversation history per user session using Redis for coherent follow-up querying in extended legal research sessions.
- Deploy a web-based interactive interface accessible to legal professionals, law students, and non-expert users requiring plain-language legal guidance.

B. System Modules

The proposed system is organized into five functional modules. The Document Processing Module loads and structures legal texts from the IPC JSON dataset, segmenting content into section-level chunks with metadata including section number and title. Each section text is formatted as a combined string of section identifier, title, and content for embedding. The Embedding and Indexing Module applies InLegalBERT through the HuggingFace Embeddings interface to generate dense 768-dimensional vector representations for each legal passage, which are then indexed in a FAISS flat L2 vector store and saved locally for persistent retrieval.

The Hybrid Retrieval Module processes each incoming user query through two parallel retrieval paths. The first path applies a regular expression to detect explicit IPC section numbers in the query and performs direct JSON lookup, returning the verbatim section text with zero semantic ambiguity. The second path performs FAISS similarity_search_with_score to retrieve the top-k semantically similar passages, applying a score threshold of 0.65 to gate retrieval and prevent irrelevant low-confidence results from contaminating the context. The retrieval source type—JSON, RAG, HYBRID, or GEN—is recorded for provenance tracking.

The Response Generation Module constructs a structured prompt combining the system persona (AI Legal Assistant specialised in Indian law), the retrieved IPC context, the last four turns of conversation history, and the current user query. This prompt is submitted to the Groq API using the configured LLM, which generates a concise, educational, and academically phrased legal response with maximum 800 tokens. A fallback retry mechanism with a softened prompt is activated if the primary

response is empty or begins with apologetic phrasing. The Conversation Management Module uses Upstash Redis to persist chat history as JSON-serialized lists of past queries, generated responses, and source types, keyed by chat session name. New sessions are created on demand and listed for session switching.

V. SYSTEM ARCHITECTURE AND DESIGN

A. Architecture Overview

The system architecture follows a five-layer pipeline. The Data Layer stores the IPC sections in a structured JSON file (laws_raw.json) organised by section number, with each entry containing a title and full content field. The Embedding Layer applies InLegalBERT via HuggingFaceEmbeddings to transform each section's text into a 768-dimensional dense vector. The Index Layer stores these vectors in a FAISS flat index (ipc_embed_db_inlegalbert) enabling GPU-accelerated or CPU approximate nearest-neighbour retrieval. The Retrieval and Generation Layer combines hybrid retrieval output with LLM generation via the Groq API. The Interface Layer provides a web frontend for user interaction with session persistence via Redis.

B. InLegalBERT Embedding Model

InLegalBERT (law-ai/InLegalBERT) is a BERT-based transformer model pretrained on a large corpus of Indian legal documents including Supreme Court and High Court judgments, IPC texts, and legislative acts. Unlike general-purpose BERT, InLegalBERT captures domain-specific legal vocabulary, citation patterns, and the formal syntactic structures characteristic of Indian legal prose. The model produces 768-dimensional contextual embeddings that encode semantic meaning at the sentence and passage level, enabling similarity-based retrieval that understands legal synonyms, related provisions, and cross-referenced sections that keyword search cannot capture.

C. FAISS Vector Store

FAISS (Facebook AI Similarity Search) provides efficient similarity search over large collections of dense vectors using optimized index structures and distance computation routines. The system uses a flat L2 index for exact distance computation, trading index construction time for retrieval accuracy. The vector store is built once from the IPC JSON data during initial setup and persisted locally, enabling rapid load on subsequent application starts. The similarity_search_with_score method returns results with associated L2 distances, enabling the score thresholding mechanism (threshold=0.65) that prevents low-confidence retrievals from entering the context prompt.

D. Hardware and Software Requirements

The system software stack comprises Python 3.10 as the primary language, LangChain Community for FAISS integration and document schema management, HuggingFace Transformers and HuggingFaceEmbeddings for InLegalBERT model loading, FAISS-cpu for vector indexing and retrieval, Groq Python SDK for LLM API access, Upstash Redis client for conversation persistence, python-dotenv for environment variable management, and a web frontend built with HTML and CSS. Hardware requirements include an Intel Core i5 or higher processor at 2.50 GHz or above, 8 GB RAM minimum, and 250 GB storage. GPU acceleration is optional and improves embedding generation throughput for large document ingestion.

VI. IMPLEMENTATION

A. Embedding and Index Construction

The build_faiss_index function loads laws_raw.json, iterates through all IPC sections, constructs a formatted text string combining section number, title, and content for each entry, and creates a LangChain Document object with the text as page_content and section metadata. The FAISS.from_documents class method generates embeddings for all documents using get_embedding_model() and builds the flat index. The completed index is saved to the local filesystem using save_local. The get_embedding_model function implements a lazy singleton pattern, instantiating InLegalBERT only once per application process to avoid redundant model loading overhead. AutoTokenizer and AutoModel from HuggingFace are pre-cached to the local model cache directory.

B. Hybrid Retrieval Logic

The hybrid_retrieve function implements a two-stage retrieval pipeline. Stage one applies re.search with the pattern \b section\s*(\d+)\b to detect explicit section references in the user query. If found, the section number is used as a key to directly retrieve the corresponding IPC entry from laws_raw.json, bypassing the vector search for this component and assigning source type "JSON". Stage two loads the FAISS vectorstore via get_vectorstore and calls similarity_search_with_score(query, k=5) to retrieve the five most semantically similar passages with their L2 distances. The top score is compared against the threshold: if below 0.65 or if the query contains legal keywords (ipc, section, article, act), all retrieved passages are added to the context and source is updated to "RAG" or "HYBRID". The final context string is the newline-joined concatenation of all collected passages.

C. Response Generation with Groq API

The process_input function orchestrates the full query processing pipeline. It loads the current session's chat history from Redis, constructs a history_prompt from the last four conversation turns, calls hybrid_retrieve for context and source type, and builds a full_prompt combining the SYSTEM_PROMPT (AI Legal Assistant persona instruction), the context or fallback message, conversation history, and the current user query. The _groq_generate_once function submits the prompt to groq_client.chat.completions.create with parameters: model=GROQ_MODEL_NAME, temperature=0.2, max_tokens=800,

top_p=0.9. A two-attempt fallback handles empty responses by retrying with a softened academic framing. The final response and source type are appended to Redis chat history via save_chat.

VII. RESULTS AND DISCUSSION

System testing was conducted across seven distinct testing levels to validate all functional and non-functional requirements. Unit testing validated individual module functions including JSON section lookup accuracy, FAISS index construction and load integrity, embedding model lazy initialization, Redis session create-load-save cycles, and Groq API response parsing. All 26 unit test cases passed, confirming correct module-level behaviour.

Functional testing evaluated the system on a set of 30 representative legal queries spanning four categories: explicit IPC section queries (e.g., "What is IPC Section 302?"), conceptual legal queries (e.g., "What constitutes culpable homicide?"), case-based queries (e.g., "What are the elements of theft under Indian law?"), and follow-up queries testing conversation continuity. Source type analysis showed that 43% of queries were resolved through direct JSON lookup, 38% through FAISS semantic retrieval, 12% through hybrid combination, and 7% through general LLM generation without retrieved context.

Table II: System Performance Metrics

Metric	Value	Notes
Average Response Time	2.4 sec	End-to-end on i5 CPU
FAISS Retrieval Time	< 50 ms	Across full IPC index
Embedding Generation	~180 ms/section	InLegalBERT, CPU
JSON Section Accuracy	100%	Exact match by section no.
Semantic Retrieval Precision	87.3%	30-query functional test
Redis Session Persistence	100%	All sessions restored
Unit Tests Passed	26 / 26	All modules validated

Semantic retrieval precision of 87.3% was evaluated by comparing retrieved passages to expected relevant IPC sections for each query, assessed by two legal domain reviewers. The hybrid retrieval mechanism improved precision by 9.2 percentage points compared to FAISS-only retrieval, demonstrating the complementary value of structured section lookup for explicit legal references. Response quality was assessed on a 5-point Likert scale for accuracy, completeness, and plain-language clarity by five law students; mean scores were 4.3, 4.1, and 4.5 respectively.

Table III: Test Case Summary

Test Type	Cases	Passed	Result
Unit Testing	26	26	Pass
Functional Testing	30	28	Pass (2 edge cases noted)
Integration Testing	12	12	Pass
Performance Testing	5	5	Pass
Security Testing	4	4	Pass
Usability Testing (UAT)	6	6	Pass

VIII. ADVANTAGES OF PROPOSED SYSTEM

- Domain-Specific Legal Understanding: InLegalBERT embeddings trained on Indian legal corpora capture legal vocabulary, citation patterns, and formal syntactic structures specific to Indian law, enabling semantically accurate retrieval that general-purpose models cannot achieve.
- Hybrid Retrieval Accuracy: The two-stage retrieval combining JSON section lookup with FAISS semantic search achieves 87.3% precision and 100% accuracy on explicit section queries, outperforming FAISS-only retrieval by 9.2 percentage points.
- Fast Response Time: Average end-to-end response time of 2.4 seconds with sub-50 ms FAISS retrieval makes the system practical for real-time legal research without dedicated GPU hardware.
- Grounded and Reliable Responses: RAG prompting ensures that generated responses are grounded in retrieved IPC sections and legal passages rather than hallucinated facts, critical for a legal assistance application where factual accuracy is essential.
- Multi-Turn Conversation Management: Redis-based session persistence maintains complete query-response history across extended research sessions, enabling coherent follow-up questions and contextual refinement of legal queries.
- Accessible Legal Information: Plain-language response generation makes IPC provisions and legal concepts accessible to non-expert users including law students, researchers, and individuals seeking self-guided legal understanding.

- Scalable Architecture: The modular pipeline supports extension to additional legal corpora, multiple jurisdictions, new transformer models, and additional LLM backends without restructuring the core retrieval or generation logic.

IX. CONCLUSION

This paper presented an AI-Based Legal Document Understanding and Case Query Assistant that integrates InLegalBERT semantic embeddings, FAISS vector retrieval, hybrid JSON and semantic search, Groq API LLM response generation, and Redis conversation persistence into a complete end-to-end legal intelligence pipeline. The system achieves 87.3% semantic retrieval precision, 100% accuracy on explicit IPC section queries, and 2.4-second average response time on standard CPU hardware, demonstrating practical viability for real-world deployment as a legal research tool. The grounded RAG architecture ensures factual reliability in responses, addressing the legal hallucination risk inherent in unconstrained LLM generation.

The system reduces legal research time significantly by replacing manual document review and keyword search with semantic understanding and generative summarization. It makes IPC provisions and legal reasoning accessible to law students, researchers, and non-expert users through plain-language responses while providing legal professionals with a rapid preliminary case analysis tool. The hybrid retrieval mechanism and domain-specific InLegalBERT embedding model represent meaningful advances over general-purpose legal AI systems deployed without Indian legal corpus adaptation.

Future enhancements will pursue five directions: (1) expansion of the legal corpus beyond IPC to include the Code of Criminal Procedure, Indian Evidence Act, and selected High Court judgment collections; (2) integration of predictive analytics to estimate likely case outcomes based on historical judgment patterns; (3) multi-language support for Tamil, Hindi, and other regional languages to improve accessibility across Indian states; (4) voice-based query interface using speech recognition and text-to-speech for hands-free legal research; and (5) blockchain-based document verification for tamper-proof authentication of uploaded legal documents in a production deployment environment.

REFERENCES

[1] M. Okumura, Y. Matsumoto, and M. Yoshikawa, "A Legal Question Answering System Using Deep Learning and Semantic Search," 2025.
[2] R. Gupta, A. Sharma, and N. Gupta, "Automated Summarization of Legal Decisions Using Transformer Models," 2025.
[3] K. Branting, T. L. Martinez, and C. A. Knoblock, "Leveraging Natural Language Processing for Legal Document Analysis," 2024.
[4] Y. Cao, Z. Zhang, and W. Liu, "Deep Learning Models for Legal Text Classification and Precedent Extraction," 2025.
[5] S. Montemagni, T. Declerck, and N. Calzolari, "Improving Legal Information Retrieval with Semantic Web and NLP Technologies," 2024.
[6] R. Udayakumar, H. M. Abbas, R. Velmurugan, A. Vanathi, and N. K. Sundaram, "Context-Aware Query Interpretation in Legal Information Systems," 2025.
[7] J. A. de Oliveira Lima, "Unlocking Legal Knowledge with Multi-Layered Embedding-Based Retrieval," 2024.
[8] M.-Y. Kim, J. Rabelo, H. K. B. Babiker, M. A. Rahman, and R. Goebel, "Legal Information Retrieval and Entailment Using Transformer-Based Approaches," 2024.
[9] R. M. Bakker, A. J. Schoevers, R. A. N. van Drie, M. P. Schraagen, and M. H. T. de Boer, "Semantic Role Extraction in Law Texts: A Comparative Analysis of Language Models," 2025.
[10] V. Naik, R. K., and P. Patel, "Enhancing Semantic Searching of Legal Documents Through LSTM-Based NER and Semantic Classification," 2024.
[11] D. K. Lee, "Natural Language Processing for Automated Legal Document Summarization," International Meridian Journal, vol. 6, no. 6, 2024.
[12] P. Srivastava, R. Singh, R. Agrawal, and A. Shahi, "LegalEase: An AI-Driven Assistant for Legal Query Resolution and Document Summarization in India," IJRASET, 2025.
[13] H. Mentzingen, N. António, and F. Bacao, "Effectiveness in Retrieving Legal Precedents: Exploring Text Summarization and Language Models," Artificial Intelligence and Law, 2025.
[14] H.-T. Nguyen, M.-K. Phi, X.-B. Ngo, V. Tran, L.-M. Nguyen, and M.-P. Tu, "Attentive Deep Neural Networks for Legal Document Retrieval," arXiv, 2022.
[15] R. V. Kulkarni, A. Agrawal, A. Vimal, R. Barde, R. Bajaj, and K. Gaddi, "Legal Case Search: An AI-Powered Legal Search Engine," Lecture Notes in Networks and Systems, vol. 1645, pp. 354–363, 2026.