NyayGraph: A Knowledge Graph Enhanced Approach for Legal Statute Identification in Indian Law using Large Language Models

Siddharth Shukla[1*] **Tanuj Tyagi**[1*] **Abhay Singh Bisht**[1]
Ashish Sharma[1] **Basant Agarwal**[2]
[1]Manipal University Jaipur, India [2]Central University of Rajasthan
siddharth.23fe10cse00812@muj.manipal.edu
tanuj.23fe10cse00715@muj.manipal.edu *

Abstract

One of the first steps in the judicial process is finding the applicable statutes/laws based on the facts of the current situation. Manually searching through multiple legislation and laws to find the relevant statutes can be time-consuming, making the Legal Statute Identification (LSI) task important for reducing the workload, helping improve the efficiency of the judicial system. To address this gap, we present a novel knowledge graph-enhanced approach for Legal Statute Identification (LSI) in Indian legal documents using Large Language Models, incorporating structural relationships from the Indian Penal Code (IPC) the main legislation codifying criminal laws in India. On the IL-TUR benchmark, explicit KG inference significantly enhances recall without sacrificing competitive precision. Augmenting LLM prompts with KG context, though, merely enhances coverage at the expense of precision, underscoring the importance of good reranking techniques. This research provides the first complete IPC knowledge graph and shows that organized legal relations richly augment statute retrieval, subject to being integrated into language models in a judicious way. Our code and data are publicly available at Github.

1 Introduction

In India, there are about 44 million pending cases in multiple courts at various levels (district, state, federal) accreting to the National Judicial Data Grid. Such a massive backlog of cases goes against the fundamental human right of fair access to justice. Automating parts of the legal workflow, such as identifying relevant statutory provisions from legal documents, can help reduce this burden by aiding judges, lawyers, and legal researchers in retrieving the right laws more efficiently.

Legal statute identification (LSI)—the task of mapping text (e.g., facts or case descriptions) to relevant statutory provisions—is a foundational subtask in law and legal NLP. Indian law poses unique challenges: statutes are long, sections cite each other, and datasets for Indian legal NLP are only recently becoming available. The IL-TUR benchmark Joshi et al., 2024 (Joshi et al., 2024), (IL-TUR: Benchmark for Indian Legal Text Understanding and Reasoning) has recently provided a standardized testbed for a number of Indian legal tasks, including LSI; we adopt its LSI split for evaluation which comprises of 100 target statutes from the Indian Penal Code (IPC), the main legislation codifying criminal laws in India.

Large Language Models (LLMs) that are solely trained on text, however, frequently lack explicit structural knowledge of the law, which results in predictions that are either ungrounded or incomplete. We fill this gap by creating a domain-specific Knowledge Graph (KG) of the Indian Penal Code (IPC) that encodes cross-references between sections sourced from the National Crime Records Bureau as well as relationships between chapters, sections, and their titles and descriptions from IPC. We incorporate this KG, an external, verifiable source of legal knowledge, into LLMs to enhance their accuracy, interpretability, and statutory identification foundation.

2 Related Work

Over the past few years, Legal NLP has been a fertile area for research. Researchers have explored different aspects of the legal domain via various tasks. Legal Statute Identification (LSI) is one of the first steps in the judicial process is finding the applicable statutes/laws based on the facts of the current situation.

Current research has started to integrate graph structures into the analysis of legal documents. Paul et al. (2022) (Paul et al., 2022a) proposed LeSICiN, a graph-based heterogeneous model for Legal Statute Identification (LSI) that represents

*Joint first authors and contributed equally to this work.

Proceedings of the Natural Legal Language Processing Workshop 2025, pages 147–156
November 8, 2025

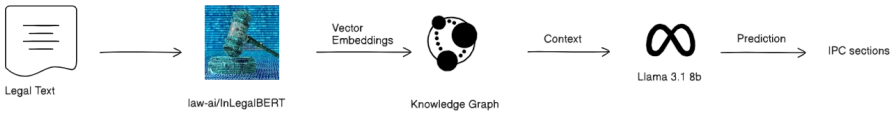

Figure 1: NyayGraph Workflow Diagram

citation networks between case documents and IPC sections. Their method integrates textual features with structural graph information utilizing metapath-based aggregation for inductive link prediction. Though LeSICiN reflects advancement in comparison to text-only approaches, it is only applicable to criminal law codes. It also requires pre-existing citation networks, and hence not viable for statutory examination in the absence of established case law precedents.

Likewise, Wendlinger et al. (2025) (Wendlinger et al., 2025) suggest mutual citation heterogeneous graph enrichment-based prediction, but with case-by-case citation instead of statutory connections.

Early methods of constructing legal knowledge graphs primarily focused on information extraction from court rulings and case files. Jain et al. (2022) (Jain et al., 2022) proposed a rule-based approach to constructing knowledge graphs from Indian Supreme Court rulings. Although Jain et al.'s work provides valuable insights into legal entity extraction, it suffers from a number of important limitations that prevent the use of the work to conduct in-depth legal analysis. Their case-focused approach identifies metadata from court decisions but does not focus on identifying the underlying statutory design or inter-section relation among legal codes.

Recent efforts have looked into various ways to model semantic relationships in legal texts. Bhardwaj et al. (2022) (Bhardwaj et al., 2022) created thematic similarity measures for Indian legal documents using knowledge graphs, focusing on capturing conceptual relationships between legal concepts rather than statutory structure.

The issue of multi-semantic relationships in legal knowledge graphs has been tackled using different embedding methods. Zhou et al.'s multi-task model incorporating translational embedding shows the importance of capturing complex semantic relationships beyond simple citation patterns (Zhou et al., 2024).

Although the majority of the current work has concentrated on case documents, scant literature has examined holistic statutory structure modeling. Conventional methods have depended greatly on citation networks and case precedents, restricting their use to full legal frameworks. Paul et al. (2022) (Paul et al., 2022a) presented LeSICiN, employing citation inter-relations between case documents and IPC sections via heterogeneous graph modeling but was limited by citation dependency.

Existing methods need either large case document sets (Dong et al., 2021; Zhou et al., 2024) or pre-existing citation networks (Paul et al., 2022a), thus limiting applicability to new or developing legal systems. Existing methods only address highly cited provisions or certain case types, and not overall statutory analysis. Existing methods are based on citation or co-occurrence relationships alone, lacking sophisticated statutory relationships like hierarchical dependencies, crime classification groupings, amendment histories etc. Existing knowledge graphs cannot be updated with the evolving legal framework and amendments, as updates need to be made by reconstructing them entirely.

In contrast to citation-based techniques or case-document centric methods, our research builds knowledge graphs from official crime statistics and statutory structure directly. By combining NCRB crime classification statistics with IPC hierarchical structure, we establish factual statistical relationships embodying actual legal practice patterns from real life. Our method overcomes important limitations of previous efforts through the ability to perform total statutory framework analysis without the need for precedential established cases, providing automated construction without legal domain expertise, utilizing a multiplicity of relationship types founded on legal structure and empirical crime evidence, and incorporating a hierarchy-agnostic framework applicable to any hierarchically structured statutory system.

3 Dataset

We compare our approach on the Legal Statute Identification (LSI) sub-task of the IL-TUR benchmark (Joshi et al., 2024), the largest and most recently available dataset for this task in the In-

Dataset Characteristic	Value
Dataset	ILSI
# Documents	66,090
# Labels	100
Train/Dev/Test Split	
Train	42,835
Dev	10,200
Test	13,039
Avg. Document Size (in #words)	2406
Avg. no. of citations (#labels per doc)	3.78

Table 2: Summary of the ILSI Dataset Statistics.

dian legal domain. It was constructed from 66,000 Supreme Court and High Court judgments, each of which cited one or more statute from a target list of 100 most frequently occurring sections of the Indian Penal Code (IPC), the primary codification law of criminal law in India. The dataset is an extension of the previous ILSI corpus released by Paul et al. (2022) (Paul et al., 2022a), with entities anonymized (e.g., `PERSON`, `ORGANIZATION`) to minimize bias, as is best practice in Indian legal NLP (Malik et al., 2021).

A full dataset statistics breakdown, including size, label distribution, and splits, is shown in Table 2.

We selected this IL-TUR LSI due to the following reasons. It possesses: (1) real-world coverage of Indian legal statutes, (2) realistic multilabel nature of data, (3) dataset size sufficient for deep learning algorithms, (4) coverage of case facts and statute descriptions, and (5) standard preprocessing and quality control. These properties are best suited for evaluating the impact of our knowledge graph-assisted approach on legal statute identification performance. The multilabel nature of the dataset—mean of 3.78 statutory citations per case—is best served by our knowledge graph design, in which interstatute relationships are explicitly modeled.

4 Methodology

4.1 Knowledge Graph Construction

We built a comprehensive domain-specific Knowledge Graph (KG) of the Indian Penal Code (IPC) through a systematic multi-source integration process. Our approach combines the hierarchical organization of the IPC statute with empirical data of crime class from actual government reports into one semantic representation of Indian criminal law.

The construction process is based solely on official government and authoritative legal publications to avoid subjective interpretation and provide accuracy. The main structural relationships of Chapters, Sections, and their definitions were taken directly from official IPC text published by the Government of India.(Crime In India 2022, Statistics Vol. I & Crime In India 2022, Statistics Vol. II). This guarantees that there is no manual interpretation or speculative legal linking in the KG and that all relationships come from reliable legal or government-published sources only.

The five node types of the KG architecture are utilized to encode different facets of legal knowledge representation. The schema is balanced between granularity and computational efficiency with full coverage and no query performance degradation. As illustrated in Table 1, the five node types are: (1) Chapter nodes for the top 26 IPC divisions, (2) Section nodes for the 571 legal provisions, (3) SectionDescription nodes for the full textual content of provisions, (4) IPC_CRIMES_HEAD nodes for the top 16 NCRB classification crime head cat-

Node Type	Count	Key Properties	Description
Chapter	26	`chapter` (unique), `chapterTitle`	Major divisions of the IPC.
Section	571	`sectionNumber` (unique), `sectionTitle`	Legal sections under each chapter.
SectionDescription	571	`id` (unique), `sectionDescription`, `embedding`	Textual content of each section and vector embedding for retrieval.
IPC_CRIMES_HEAD	16	`name`	NCRB top-level crime categories.
IPC_CRIMES_SUBHEAD	41	`name`	NCRB subcategories under each crime head.

Table 1: Node Types and Properties in the Knowledge Graph.

egories, and (5) IPC_CRIMES_SUBHEAD nodes for the 41 domain-specific crime sub-head categories. Each node type has some properties optimized for various query patterns and downstream applications.

Eight relationship types were systematically extracted from the source documents, each with specific semantic roles in legal analysis (Table 3). Hierarchical relationships (BELONGS_TO, HAS) mirror directly the statutory organization from the IPC structure. Crime classification relationships (HAS_SECTION, COMES_UNDER, HAS_SUB_HEAD) map the empirical NCRB taxonomy onto statutory provisions, enabling analysis of real crime patterns. Cross-reference relationships (CITES, CITED_IN) were extracted through systematic parsing of statutory text for overt section references. Content relationships (IS_STATED_IN) link sections to their descriptive text for semantic processing. The overall graph has 1,225 nodes linked by 2,632 relationships (Table 4), the most comprehensive structural representation of the IPC to date.

The graph was built with Neo4j due to its Cypher query language that is feature-rich and graph-traversal-optimized performance. Data ingestion employed a manual-to-digital conversion process: (1) systematic transcription of IPC hierarchical structure as explicitly organized in the official statute, (2) direct mapping of NCRB crime categories to published categorization sections without interpretation, (3) manual identification and encoding of cross-references since they evidently appear in the statutory text, and (4) automatic embedding generation of transcribed textual content with the InLegalBERT model (Paul et al., 2022b).

Metric	Value
Total Nodes	1,225
Total Relationships	2,632
Node Types	5
Relationship Types	8
Graph Database Platform	Neo4j

Table 4: Overall Knowledge Graph Summary.

All relationship construction was done using ad-hoc Cypher queries that encode the factual relationships directly from the source documents. For instance, NCRB (Table 3) clearly states which IPC sections come under each head of crime, and these mappings were translated directly into HAS_SECTION and COMES_UNDER relationships without any legal analysis or interpretation. The construction process is fully traceable to source documents, where each relationship type is traceable to particular tables or sections in the quoted government reports. This manual but objective process delivers precision without the vagaries of automated legal text parsing, which would be subject to advanced natural language processing and possible legal interpretation.

5 Evaluations and Results

5.1 KG Inferencing

To enable semantic similarity calculations, section descriptions were converted to dense vector representations via the law-ai/InLegalBERT (Paul et al.,

Relationship	From Node(s)	To Node(s)	Count	Purpose
BELONGS_TO	Section	Chapter	572	Maps each section to its chapter.
HAS	Chapter	Section	572	Hierarchical containment from chapters to sections.
HAS_SECTION	IPC_CRIMES_HEAD / IPC_CRIMES_SUBHEAD	Section	228	Links crime categories to sections.
HAS_SUB_HEAD	IPC_CRIMES_HEAD	IPC_CRIMES_SUBHEAD	41	Links NCRB crime head to its subheads.
COMES_UNDER	Section	IPC_CRIMES_HEAD / IPC_CRIMES_SUBHEAD	228	Maps sections to NCRB crime categories.
CITES	Section / Chapter	Section / Chapter	210	Indicates statutory cross-references.
CITED_IN	Section / Chapter	Section / Chapter	210	Reverse direction of CITES relationship.
IS_STATED_IN	Section	SectionDescription	571	Links section to its descriptive text.

Table 3: Relationship Types in the Knowledge Graph.

2022b) model. The transformer model was selected due to its established performance on Indian legal texts with computational efficiency in the processing of big data. The embeddings enable semantic similarity calculations across legal provisions regardless of structural connections, supporting intricate query patterns such as concept-based section retrieval and thematic clustering of similar legal provisions.

We assess the value of our IPC knowledge graph by performing a direct, graph-only inference pipeline initially that identifies applicable statutes for every case fact based on a mix of semantic similarity and graph traversal. The experiments are performed on the IL-TUR LSI test split.

We calculate 768-dimensional embeddings for every case fact using the law-ai/InLegalBERT model with mean pooling. We then index a query to the Neo4j vector index for the `SectionDescription.embedding` property to find the top-k most similar sections, where $k \in \{5, 8, 10\}$.

From the initially retrieved sections, we expand predictions by traversing three relationship types in the KG:

- *Forward citations:* Sections cited by the retrieved sections (`CITES`).
- *Reverse citations:* Sections that cite any of the retrieved sections (`CITED_IN`).
- *Crime-head adjacency:* Sections sharing the same NCRB crime head or subhead (`COMES_UNDER`, `HAS_SECTION`).

Filtering and Aggregation. We normalize all predicted section labels to canonical form (e.g., "294(b)" → "294B") and filter against the 100 valid IPC sections as per IL-TUR. The final prediction set is the union of similarity and expansion candidates, with a fallback to the top-3 similarity hits when no candidates remain.

We evaluate retrieval performance in terms of a set of ranking and multi-label metrics across the IL-TUR test set. For a given test instance, we match the ranked list of predicted sections with the ground-truth set and calculate:

- **Mean Reciprocal Rank (MRR)** – The average reciprocal of the rank at which the first correct section appears (Voorhees, 1998).
- **Mean Average Precision (MAP)** – The mean of the average precision values over all test cases (Manning et al., 2008).
- **Precision@k (P@k)** – The fraction of correct sections within the top-k predictions.

$$P@k = \frac{|\{Relevant \cap Retrieved@k\}|}{k}$$

- **Recall@k (R@k)** - The fraction of true sections retrieved in the top-k.

$$R@k = \frac{|\{Relevant \cap Retrieved@k\}|}{|\{Relevant\}|}$$

(Manning et al., 2008).

- **Normalized Discounted Cumulative Gain@k (NDCG@k)** – position-weighted measure of ranking quality, normalized by the ideal DCG (Järvelin and Kekäläinen, 2002).
- **Hit@k (H@k)** - The percentage of cases with at least one true section in the top-k predictions (Manning et al., 2008).

The figure in Table 5 illustrates the effectiveness of our knowledge graph-augmented method for Legal Statute Identification at different retrieval depths. Some surprising insights can be deduced from this comparison:

Scaling of Performance with k: All the performance metrics improve steadily as k scales from 5 to 10. MRR improves by 43.3% (0.0826 → 0.1184), i.e., more relevant sections get ranked higher in larger result lists. Similarly, MAP improves by 42.3% (0.0286 → 0.0407), i.e., precision on all relevant items is improved. This scaling trend indicates that the graph traversal effectively

Run (k)	MRR	MAP	H@k	P@k	R@k	NDCG@k
top_k = 5	0.0826	0.0286	0.1073	0.0237	0.0380	0.0424
top_k = 8	0.1038	0.0359	0.1901	0.0278	0.0672	0.0586
top_k = 10	**0.1184**	**0.0407**	**0.2538**	**0.0309**	**0.0899**	**0.0709**

Table 5: KG inference performance on the IL-TUR test set for different top_k values.

retrieves more relevant sections beyond the initial similarity-based retrieval.

Enhancement in Recall through Graph Traversal: Most significant enhancements are in recall metrics. R@10 is 0.0899, which is a 136.8% enhancement over R@5 (0.0380). notable recall enhancement supports our hypothesis that structural relationships in the IPC knowledge graph capture relevant connections between statutes not apparent through text similarity. The crime-head adjacency and citation relationships correctly identify relevant provisions with shared legal contexts.

Hit Rate and Coverage: Hit@k improves dramatically to 25.38% at k=10, It is a 136.5% improvement on Hit@5 (10.73%), which means that graph expansion greatly improves the chances of returning relevant statutes for all cases.

Precision-Recall Trade-off: Precision sees only slight improvements (P@5: 0.0237 $\rightarrow$ P@10: 0.0309), whereas the dramatic recall improvements confirm that our approach does expand the relevant candidate set without too much spuriousness. The NDCG@k improvements (0.0424 $\rightarrow$ 0.0709) confirm that the additional retrieved segments also have good ranking quality.

Challenges in Legal Domain: The multi-label aspect of the task, with an average of 3.78 labels per case in IL-TUR, means that even modest gains in each measure have significant practical utility in legal scholarship and case analysis.

These experiments show that our IPC knowledge graph is highly effective for statute identification, with the graph traversal component having substantial recall and coverage enhancements while maintaining competitive precision. The steady enhancement of all the measures as k grows larger shows that practitioners can adjust the depth of retrieval to their specific precision-recall needs.

5.2 LLM Inference

We built and evaluated a retrieval-augmented LLM pipeline that improves a Large Language Model with structured context from the IPC Knowledge Graph (KG). The setup uses a Neo4j vector index for semantic retrieval and an Ollama-hosted Llama3.1 8B model (Grattafiori et al., 2024) for scoring and generation. Detailed configurations of the inference pipeline is listed in Table 6. The completed scripts are available in the supplementary repository.

5.2.1 Pipeline

Given case facts, the LLM pipeline executes the following steps:

1. **Semantic retrieval:** Encode the input using a legal-domain Bert based Transformer model (`law-ai/InLegalBERT`, mean-pooling, 512-token truncation) and query the Neo4j vector index `section_desc_embedding_index` to retrieve the top-k section description nodes (default $k = 3$). InLegalBERT is a legal-domain PLM shown to improve performance on Indian legal tasks. (Paul et al., 2022b)

2. **KG expansion:** For each retrieved section the system retrieves (i) outbound cited sections (`CITES`), (ii) inbound citations (`CITED_IN`), and (iii) other sections under the same NCRB crime head/subhead (`COMES_UNDER`, `HAS_SECTION`). These Cypher queries are executed in the retriever class and returned in a structured context object.

3. **Prompt construction:** The KG-formatted context is added to the case facts. A limited system instruction then tells the model to output *only* canonical IPC section numbers in a bracketed list (this reduces hallucination and

Hyperparameter / Setting	Value (from code)
KG embedding model	`law-ai/InLegalBERT`
Embedding pooling	mean-pooling, max_len=512
Neo4j vector index	`section_desc_embedding_index`
KG retrieval top-k	3
Ollama temperature (example run)	0.6
Ollama max tokens (example run)	4096
Dataset	`Exploration-Lab/IL-TUR`, subset=`lsi` (test split)
Prompt format	KG context + Case facts + restricted system prompt

Table 6: Configurations used for the LLM+KG inference runs (values taken from the provided scripts).

simplifies automatic evaluation). Approaches that combine text and graph structure for LSI have shown to be effective in prior work (e.g., LeSICiN). (Paul et al., 2021). Refer A.1 for system prompt.

4. **LLM scoring/generation:** The enhanced prompt is sent to the Ollama generation API, the response and the KG context used are saved for each example.

5. **Post-processing:** model outputs are normalized to canonical section tokens (e.g., `302`, `294B`), mapped to the IL-TUR 100-section target set (Joshi et al., 2024), and added to a CSV for evaluation.

We evaluate model outputs using a deterministic post-processing and metric pipeline.

5.2.2 Normalization & mapping

The evaluation pipeline performs three main steps:

1. **Normalize raw model responses** (function `normalize_model_response`): this extracts numeric tokens and common suffixes (A/B/C), expands numeric ranges (e.g., '402-405' → '402 403 404 405'), collapses tokens like '403 (a)' → '403A', strips noise words (e.g., 'section', 'ipc'), and returns a canonical bracketed string (e.g., '`[302 304]`').

2. **Map canonical tokens to IL-TUR IDs:** a reverse mapping converts normalized section tokens into the IL-TUR label ids (1..100). We preserve the canonical format used by the IL-TUR benchmark to avoid label-mismatch issues. (Joshi et al., 2024)

5.2.3 Evaluation Metrics

Outputs are binarized with `sklearn`'s `MultiLabelBinarizer` and evaluated using micro/macro Precision, Recall and F1; per-sample precision/recall/F1 are also computed and appended to the CSV for fine-grained analysis. (Pedregosa et al., 2011). We report both **micro-averaged** and **macro-averaged** Precision, Recall, and F1 scores. Micro scores treat all true and predicted section labels across the test set as one group. This highlights overall correctness. Macro scores average the metrics across classes. They give equal weight to both rare and frequent sections.

5.2.4 Quantitative comparison and analysis

Table 7 summarizes micro- and macro-averaged Precision, Recall, and F1 for three inference modes: (i) a standard LLM baseline, (ii) the LLM enhanced with IPC KG context (LLM+KG, top_k=10), and (iii) a KG-only expansion-based retrieval. The standard LLM achieves the highest overall micro-F1 score of 0.072, showing the best balance between precision and recall under strict multi-label evaluation. The KG-only pipeline attains the highest recall (micro R = 0.091), but this comes with very low precision, resulting in many false positives. The combined LLM+KG approach improves recall compared to the standard LLM (0.067 vs 0.061), but it results in a lower micro-F1 score of 0.059. This indicates that the model did not filter or re-rank the additional candidates from the KG effectively.

These results suggest that while the KG greatly improves coverage by reducing false negatives, simply adding KG context to the prompt or making basic expansions increases false positives. It requires a stronger re-ranking or calibration step to turn this coverage into improved accuracy. Retrieval-augmented methods typically need a learned re-ranking or calibration stage to transform recall gains into better overall accuracy (Lewis et al., 2020; Nogueira and Cho, 2019; Karpukhin et al., 2020).

Why did the KG not uniformly improve F1? Our analysis points to several factors:

- **Precision-recall tradeoff from KG expansion.** The KG-only expansion significantly boosts candidate recall but also adds many irrelevant candidates. Without a good reranker,

Run	P_{micro}	R_{micro}	$F1_{micro}$	P_{macro}	R_{macro}	$F1_{macro}$
Vanilla_Tnference_Llama_3.1_8B	**0.087**	0.061	**0.072**	**0.082**	0.066	**0.048**
KG_Inference_top_k_10_Llama_3.1_8B	0.053	0.067	0.059	0.071	0.074	0.035
KG_Only_Inference_top_k_10	0.044	**0.091**	0.059	0.028	**0.098**	0.021

Table 7: LLM and KG inference results on the IL-TUR LSI test split. Values are micro- and macro-averaged Precision, Recall and F1. The KG runs use expansion with `top_k=10` (chosen from prior KG-only tuning).

the LLM is more likely to include those irrelevant candidates in its output (Sokolova and Lapalme, 2009; Powers, 2011).

- **Class imbalance.** The low macro-F1 values indicate many classes are under-served. KG expansion can increase macro-recall in some cases, but it results in poor macro-precision (Sokolova and Lapalme, 2009).
- **Prompt and decoding effects.** Prompt design and decoding settings (temperature, sampling strategy) significantly impact whether the added context helps or confuses the model. Deterministic decoding and few-shot prompt examples can reduce formatting and hallucination errors (Brown et al., 2020).

Practical recommendations and next ablations To turn KG coverage gains into net performance improvement we recommend the following experiments (or future work):

1. **Tune retrieval k for LLM+KG separately**: the k selected from KG-only experiments (10) may be too large when KG context is fed to the LLM. Report LLM+KG results for $k \in \{3, 5, 8, 10\}$ (Lewis et al., 2020)..
2. **Reranker**: train a learned reranker that combines the LLM score with KG-derived features (citation-degree, shared crime-head flag, shortest-path length). This should reduce false positives introduced by expansion (Nogueira and Cho, 2019; Karpukhin et al., 2020).
3. **Prompt engineering:** test deterministic inference (temperature=0), and add 1–2 few-shot examples of correct bracketed outputs to reduce format and hallucination errors (Brown et al., 2020)..
4. **Ablate KG components:** inject only citations, only NCRB crime-head context, or both; compare effects on precision/recall.

Resource constraints Our experimental scope was limited by the computational and financial resources we had. Specifically, we could not perform extensive hyperparameter sweeps, evaluate additional large instruction-tuned models, or test proprietary cloud-hosted LLMs, such as GPT-4, because of costs and infrastructure issues (Strubell et al., 2019; Schwartz et al., 2020). When possible, we prioritized controlled comparisons among vanilla LLM, LLM+KG, and KG-only using locally available Ollama-hosted models and smaller LLM families. These limitations also led us to focus the KG on the IPC instead of creating a larger multi-act KG. We chose an efficient embedding model, `InLegalBERT`, that balances retrieval performance with computational cost. We acknowledge this limitation and provide sanitized code and exact configuration details to help groups with larger computing budgets reproduce our work.

6 Conclusion

In summary, this work contributes (1) a reproducible IPC knowledge graph anchored in official statutory and NCRB sources, (2) an interoperable retrieval + KG + LLM pipeline for LSI, and (3) an empirical analysis showing that KG-derived structure meaningfully increases coverage but requires careful retrieval/ranking design to improve end-to-end statutory identification performance. We believe the KG and the experimental recipe provided here can serve as a foundation for future work in KG-grounded legal NLP, especially for targeted reranking, human-in-the-loop validation, and scalable extensions across additional Indian statutes and case-law corpora.

7 Limitations

Our study presents two principal limitations. First, the knowledge graph's scope is restricted to the Indian Penal Code and NCRB crime classifications, excluding other statutes, procedural codes, or case-law citations, which limits generalizability to civil law, regulatory frameworks, or multi-statutory contexts. Second, computational constraints necessitated the use of locally-hosted Ollama models and smaller transformer architectures, precluding evaluation of large proprietary instruction-tuned models (e.g., GPT-4, Claude) that may exhibit different performance characteristics and limiting the scope of ablation studies.

8 Future Work

A number of promising avenues follow from our results and limitations. We intend to train lightweight learned rerankers that incorporate LLM scores and KG-extracted features (citation degree, crime-head relationships, graph distances) to minimize false positives from naive expansion, while expanding the knowledge graph to include other statutes

(CrPC, Evidence Act), case citations, and amendment histories with expert verification. Systematic ablation experiments will fine-tune retrieval parameters, context crafting, and prompt engineering techniques on bigger instruction-tuned models and cloud APIs to determine strong operating points. They will also evaluate prediction soundness and explainability advantages through human-in-the-loop experiments involving legal professionals, along with hybrid retrieval architecture mixing sparse and dense approaches with reranking pipelines to enhance candidate accuracy prior to graph-based expansion.

9 Ethics Statement

All KG content and evaluation data are derived from publicly available sources (the IPC statute text and NCRB reports) and the anonymized IL-TUR benchmark; we do not use private or unredacted court records. The KG is a factual transcription of those sources and is not a substitute for legal interpretation. Outputs from our models should *never* be treated as legal advice; they are intended for research and decision-support under expert supervision only.

We take several practical mitigations: (i) preserve provenance for KG edges, (ii) use deterministic evaluation and conservative post-processing to reduce spurious matches, (iii) omit any private credentials from released artifacts, and (iv) recommend human-in-the-loop validation (legal experts) before any operational use. Finally, we acknowledge limitations (class imbalance, extraction noise, compute constraints) and encourage future work on expert audits, reranking, and controlled deployments prior to real-world use.

References

Shounak Bhardwaj et al. 2022. Knowledge graph-based thematic similarity for indian legal documents. In *Proceedings of the 19th International Conference on Natural Language Processing*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Biao Dong, Haoze Yu, and Haisheng Li. 2021. A knowledge graph construction approach for legal domain. *Tehnički vjesnik*, 28(2):357–362.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Sarika Jain, Pooja Harde, Nandana Mihindukulasooriya, Sudipto Gosh, Ankush Bisht, and Abhinav Dubey. 2022. Constructing a knowledge graph from indian legal domain corpus. In *TEXT2KG/MK@ ESWC*, pages 80–93.

Kalervo Järvelin and Jyrki Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–444.

Abhinav Joshi, Shounak Paul, Akshat Sharma, Pawan Goyal, Saptarshi Ghosh, and Ashutosh Modi. 2024. IL-TUR: Benchmark for Indian legal text understanding and reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11460–11499, Bangkok, Thailand. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of EMNLP*, pages 6769–6781.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, and Sebastian Riedel. 2020. Retrieval-augmented generation for knowledge-intensive nlp. In *Advances in Neural Information Processing Systems (NeurIPS) Workshop on Learning with Limited Labeled Data*. RAG model; retrieval+generation framework.

Vijit Malik, Rishabh Sanjay, Shubham Kumar Nigam, Kripa Ghosh, Shouvik Kumar Guha, Arnab Bhattacharya, and Ashutosh Modi. 2021. Ildc for cjpe: Indian legal documents corpus for court judgment prediction and explanation. *arXiv preprint arXiv:2105.13562*.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge university press.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Shounak Paul, Pawan Goyal, and Saptarshi Ghosh. 2021. Lesicin: A heterogeneous graph-based approach for automatic legal statute identification from indian legal documents. In *arXiv preprint arXiv:2112.14731*.

Shounak Paul, Pawan Goyal, and Saptarshi Ghosh. 2022a. Lesicin: A heterogeneous graph-based approach for automatic legal statute identification from indian legal documents. In *Proceedings of the*

AAAI conference on artificial intelligence, volume 36, pages 11139–11146.

Shounak Paul, Arpan Mandal, Pawan Goyal, and Saptarshi Ghosh. 2022b. Inlegalbert: Pre-trained language models for indian legal texts. *arXiv preprint arXiv:2209.06049*.

Fabian Pedregosa, Gérald Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Andreas Müller, and et al. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

David M. W. Powers. 2011. Evaluation: From precision, recall and f-measure to roc, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1):37–63.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2020. Green ai. *Commun. ACM*, 63(12):54–63.

Marina Sokolova and Guy Lapalme. 2009. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3645–3650.

Ellen M Voorhees. 1998. Variations in relevance judgments and the measurement of retrieval effectiveness. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 315–323.

Lorenz Wendlinger, Simon Alexander Nonn, Abdullah Al Zubaer, and Michael Granitzer. 2025. The missing link: Joint legal citation prediction using heterogeneous graph enrichment. *arXiv preprint arXiv:2506.22165*.

Jie Zhou, Xin Chen, Hang Zhang, and Zhe Li. 2024. Automatic knowledge graph construction for judicial cases. *arXiv preprint arXiv:2404.09416*.

A Appendix

A.1 System Prompt

This system prompt is used in both vanilla and KG enhanced LLM inferencing.

```
You are an intelligent Legal Crime Classification system. In the Indian legal system, the Indian Penal Code (IPC) is an Act in the Indian legislature that contains many legal articles or 'Sections' that codify different laws. Your task is, given the facts or evidence of an Indian court case as input, to predict the relevant or violated 'Sections' of the IPC as output.
Only predict from the following IPC Sections:
Section 2, Section 3, Section 4, Section 5, Section 13, Section 34, Section 107, Section 109, Section 114, Section 120, Section 120B, Section 143, Section 147, Section 148, Section 149, Section 155, Section 156, Section 161, Section 164, Section 173, Section 174A, Section 186, Section 188, Section 190, Section 193, Section 200, Section 201, Section 228, Section 229A, Section 279, Section 294, Section 294(b), Section 299, Section 300, Section 302, Section 304, Section 304A, Section 304B, Section 306, Section 307, Section 308, Section 313, Section 320, Section 323, Section 324, Section 325, Section 326, Section 332, Section 336, Section 337, Section 338, Section 341, Section 342, Section 353, Section 354, Section 363, Section 364, Section 365, Section 366, Section 366A, Section 375, Section 376, Section 376(2), Section 379, Section 380, Section 384, Section 389, Section 392, Section 394, Section 395, Section 397, Section 406, Section 409, Section 411, Section 415, Section 417, Section 419, Section 420, Section 427, Section 436, Section 437, Section 438, Section 447, Section 448, Section 450, Section 452, Section 457, Section 465, Section 467, Section 468, Section 471, Section 482, Section 494, Section 498, Section 498A, Section 500, Section 504, Section 506, Section 509, Section 511
Your output MUST be ONLY the list of relevant IPC Section numbers in square brackets, separated by spaces.
Do NOT include any explanation, punctuation, or text other than this list format.
```