

Augmented Question-guided Retrieval (AQgR) of Indian Case Law with LLM, RAG, and Structured Summaries

Vishnuprabha V

vishnuprabha@cusat.ac.in

Cochin University of Science and Technology

Daleesha M Viswanathan

Cochin University of Science and Technology

Rajesh R

Naval Physical & Oceanographic Laboratory

Aneesh V Pillai

Cochin University of Science and Technology

Research Article

Keywords: Case Law Retrieval, Structured Summary Generation, Large Language Model, Retrieval Augmented Generation, Indian Legal NLP, Legal Issue Generation

Posted Date: October 13th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10487972/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Augmented Question-guided Retrieval (AQgR) of Indian Case Law with LLM, RAG, and Structured Summaries

Abstract

Finding the right precedent is one of the slowest parts of legal work, especially in India, where judgments are long and unstructured. Traditional retrieval is fact-based: it ranks precedents by similarity match and rarely explains why a case showed up, leaving the lawyer to read through each ranked judgment just to uncover the legal issue and build a proper argument. The proposed AQgR (Augmented Question-guided Retrieval) framework instead generates the legal issues a case raises and uses those issues as the query to retrieve candidate precedents through an ensemble retriever; an LLM then reranks the resulting pool using a combination of the case facts and the generated issues. Judgments are compressed into short, structured summaries via Retrieval-Augmented Generation (RAG) to fit the LLM’s context window, giving the model something to reason over rather than skim. We tested this in two stages on the FIRE 2019 AILA benchmark of Indian Supreme Court judgments. Gemini Pro, on 50 judgments, scored a MAP of 0.365 and recall of 0.672. A larger study with Llama 3 70B across about 500 judgments compared query types, summary formats, and re-ranking methods. Issue-based retrieval beat fact-based retrieval overall, and the best setup reached a MAP of approximately 0.312 (run-to-run variance up to ± 0.03 –0.04), well above the 0.1573 MAP on the FIRE 2020 leaderboard, though the setups aren’t directly comparable. The LLM also writes a one-line relevance reason per case, letting a lawyer decide whether the full judgment is worth reading.

Keywords: Case Law Retrieval, Structured Summary Generation, Large Language Model, Retrieval Augmented Generation, Indian Legal NLP, Legal Issue Generation

1 Introduction

Legal professionals worldwide, particularly in nations with common law or mixed legal systems, rely heavily on case law retrieval as a core part of their practice. India operates under a hybrid legal framework that combines common law principles with written statutes to resolve legal disputes. Common law, rooted in past judicial decisions (precedents), differs from statutory law, which consists of legislated rules and regulations, including those protecting a citizen’s fundamental rights.

Locating relevant precedents has a major bearing on a lawyer's success in a case . An experienced lawyer who has argued many trials develops a sense for how a judge is likely to frame the issues given a set of facts. Drawing on that experience and historical knowledge, the lawyer then searches for principles, drawn from precedents and statutes, that can be applied to the issue to secure a remedy for the client. Locating those principles takes both expertise and time, and since lawyers typically bill by the time spent on a case, there is a clear practical case for an automated case law retrieval system to support this work.

Automated case law retrieval systems hold real promise for helping early-career legal professionals build stronger arguments by surfacing relevant precedents [Ashley \(1992\)](#). Judges, too, could use such a tool to support precedent-based reasoning and deliver judgments more quickly. Faster decision-making of this kind reduces case backlogs and improves access to justice more broadly.

The Appellant is a member of Parliament, having been elected to the 17th Lok Sabha from Ghazipur constituency in 2019. He suffered a conviction Under Section 3(1) of the Uttar Pradesh Gangsters and Anti-Social Activities (Prevention) Act, 1986 ("Gangsters Act", hereafter) vide judgment of the Special MP/MLA Court, Ghazipur ("Trial Court", hereafter) dated 29th April, 2023, consequent where to he was sentenced to four years imprisonment together with a fine of Rupees One lakh. Aggrieved thereby, he approached the High Court for suspension of execution of the sentence as well as for suspension of the order appealed against which has succeeded in part as noted above. The refusal of the High Court to stay the conviction of the Appellant has resulted in his disqualification from the membership of Parliament by operation of law, i.e., Section 8(3) of the Representation of the People Act, 1951 ("the RoP Act", hereafter), which has duly been notified by the Lok Sabha Secretariat. As a sequel thereto, the Appellant stands barred from partaking in the electoral process for six years from the date of serving his sentence.

Can an appellate court refuse to suspend a sentence based solely on the refusal of the accused's advocate to argue the appeal?

The Appellant-Accused was convicted by the Trial Court for the offence punishable Under Section 6 of the Protection of Children from Sexual Offences Act, 2012 (for short 'POCSO Act'). The appeal against conviction preferred by the Appellant was admitted by the High Court and by the order dated 12th January, 2018, the substantive sentence of the Appellant was suspended and he was ordered to be enlarged on bail. On 7th July, 2023, the said Criminal Appeal of the year 2017 was called out before the learned Single Judge of the High Court for hearing. The Advocate for the Appellant sought adjournment for four weeks. Only on the ground that the Appellant is enjoying the facility of bail and that his advocate applied for adjournment, the High Court proceeded to cancel the bail.

Fig. 1 Figure shows the example of a legal issue with two different fact scenarios

Unstructured and inconsistent judgment formatting.

In India, the sheer volume of unstructured judgments, together with the absence of consistent headings or separation between rhetorical roles such as facts and issues, makes this automation difficult. Existing approaches to Indian precedent retrieval rank a judgment’s relevance to a given fact pattern using little more than a similarity score, which means a lawyer still has to read the full judgment to work out which legal principle actually applies.

Reliance on facts rather than legal issues.

A further limitation of current systems is their reliance on facts rather than legal issues. In practice, lawyers identify relevant case law primarily through the legal issue at stake, not through the underlying facts, and two cases with very different facts can still turn on the same legal issue. Figure 1 illustrates one such case. Fact-based similarity retrieval therefore tends to surface judgments with similar facts rather than judgments that share the same legal issue. Finding the legal issue in a fact pattern is challenging because it requires legal reasoning and knowledge. Generative AI models can help by examining the facts and producing relevant legal questions.

Lack of explanation for retrieved precedents.

Existing automated systems for retrieving Indian precedents fall short of conveying why a precedent is relevant or how it applies to a given fact pattern. This work addresses that gap by drawing on the ability of Large Language Models (LLMs) Radford et al. (2018) to generate context-aware text. An automated case law retrieval system that also surfaces the applicable legal principle could save lawyers a substantial amount of time and effort.

Notwithstanding these limitations, this work focuses on the practical utility of LLMs for explainable case law retrieval, on the expectation that usage costs will fall and context windows will grow over time. It also examines the value of structured, template-based summaries for this task. A template-like summary makes it possible to search more precisely across facts, statutes, precedents, legal issues, and other fields, yet almost no such templates currently exist for Indian legal judgments. Building one with conventional deep learning would normally require a separate model for each legal concept to be extracted, whereas an LLM trained across many tasks is naturally suited to generating template-based summaries directly. Producing these structured summaries with human legal experts is costly and slow, particularly for long judgments, so an LLM-generated alternative is attractive option. Retrieval-Augmented Generation (RAG) Lewis et al. (2020) is a framework for integrating an external knowledge source with an LLM without fine-tuning it on domain-specific data, and is used here both to generate structured summaries and to support case law retrieval by surfacing only the content that fits within the model’s context window.

Accordingly, this research addresses the following questions:

- How well do LLMs perform case law retrieval under context length restrictions and without fine-tuning?
- How do structured summaries contribute to LLM-based case law retrieval?

- What does using the legal issue, rather than the facts, contribute to case law retrieval?
- Can Retrieval-Augmented Generation (RAG) help LLMs manage lengthy legal documents and reduce hallucination?
- How well can RAG generate structured, template-like summaries from a single query under zero-shot learning?

The contributions of this work to the legal and AI communities are as follows.

- **Query document density alignment as a retrieval principle:** For similarity based retrieval, especially with lexical methods like BM25, how detailed or sparse the document representation is matters more than what type of query is used. If the query is short and compact, like a list of legal issues, it works best when matched against an equally compact document. If the query is long and descriptive, it needs a document that is just as rich in detail. Getting this balance right turns out to matter more than simply choosing the best sounding query type on its own.
- **A citation-content finding:** The citation information improves LLM reranking only when it states the legal principle behind a citation, not when it lists the citation’s name alone or its full citation details; the full citation block consumes context space the LLM could otherwise use for retrieval-relevant content and adds noise rather than signal, while a format that states the principle behind each citation produces the best result in the study.
- **An approach for case law retrieval using LLMs without fine-tuning:** A retrieval and reranking pipeline that uses large language models without fine-tuning, providing relevance explanations alongside retrieved cases.
- **A practical route past LLM context length limits:** We show that reducing a judgment to a small set of retrieval-relevant fields lets far more of the candidate pool fit inside a single context window at once. Keeping just six fields, facts, issues, reasoning (or citation content), new legal principle, and important subjects, lets 227 of our 496 judgments fit within context simultaneously, compared to only 98 when all 21 fields are kept, making LLM-based retrieval and reranking practical over a pool this size without needing a larger context window or a smaller candidate pool.
- **A systematic ablation across query, pooling, and document design choices:** Across seven experiments and over 50 queries, we vary query formulation (facts, issues, or facts-plus-issues; short versus long issue phrasing), retrieval pooling (a single query-type pool versus a union pool fused across query types), document representation (plain text versus four structured JSON formats differing in which fields are kept), and re-ranking prompt design, to identify which of these choices actually drives retrieval and re-ranking quality, and which interact with one another rather than acting independently.

2 Related Works

In legal research, case law retrieval and case law entailment have typically been treated as separate tasks. The COOLIE competition, discussed by Kim et al. (2023) [Kim](#)

et al. (2023), stands out for covering four distinct legal tasks: case law retrieval, case law entailment, statute retrieval, and statute entailment. Case law retrieval involves searching and retrieving relevant legal cases or precedents from a large repository of legal documents based on a query. Case law entailment, by contrast, asks whether a given legal precedent (the source case) entails the outcome of another case (the hypothesis case); this means finding the specific paragraph of the retrieved case that applies to the facts at hand, which could be a reasoning paragraph, a legal-issue paragraph, or a fact paragraph.

A comparable competition exists for the Indian legal context. FIRE 2019, described by Bhattacharya et al. (2019) Bhattacharya et al. (2019), focused on two tasks, case law retrieval and statute retrieval. The subsequent FIRE 2020 competition (Bhattacharya et al., 2020) Bhattacharya et al. (2020) added rhetorical role labelling to address the unstructured nature of Indian judgments. The best mean average precision (MAP) reported for precedent retrieval was 0.1573 in FIRE 2020 Leburu-Dingalo et al. (2020) and 0.1492 in FIRE 2019.

Sampath et al. (2022) Sampath and Durairaj (2022) report a notably higher MAP of 0.632 for Indian case law retrieval, but their approach uses whole judgments as queries to find related judgments, which differs fundamentally from fact-based retrieval. This is effective for finding similar legal documents, but less practical in real-world use, where lawyers typically work from factual statements or legal questions rather than full prior judgments. The work by Mandal et al. (2021) Mandal et al. (2021) similarly measures similarity between legal documents rather than retrieving against a fact pattern. Table 1 shows that almost all of this prior work uses either a legal fact or a full legal judgment as the input query for retrieval or entailment. Most fact-based retrieval work in the Indian setting finds relevant case law by computing the similarity between the embedding of a given fact pattern and the embeddings of judgments in the pool, treating the highest-scoring judgments as most relevant. Because embeddings in a typical deep learning framework have a fixed dimension regardless of input length, a roughly 1,000-word fact pattern and a 20,000-word judgment end up represented in exactly the same space, which discards a great deal of information and can degrade retrieval quality. The most relevant judgments identified this way are then typically passed on to an entailment step, which takes the fact and the candidate judgment as input and returns the paragraph that most directly entails the given fact.

A recent and closely related system, Nigam et al. (2025) Nigam et al. (2025), also moves away from using full judgments as queries. Their system classifies sentences already present in the query case into rhetorical roles, such as Facts, Issue, Arguments, Reasoning, and Decision, using a trained classifier, and combines the Facts and Issue segments as the retrieval query, with results fused across BM25, dense retrieval, and cross-encoder reranking. They find that combining facts, issue, and reasoning segments consistently beats a facts-only query, on both an Indian (IL-PCR) and a Canadian (COLIEE) corpus. This supports the same idea this paper relies on, that an issue-level signal captures relevant precedent a facts-only query misses, but reaches it differently: their issue segment is extracted from text already present in the query case by a trained classifier, not generated by an LLM, and their pipeline does not use an LLM at any stage or explain why a case was retrieved.

Table 1 Relevant related works on case law retrieval.

Author	Cita- tions	Task	Input	Output	Jurisdiction
Vuong et al. Vuong et al. (2023)		Case Law Retrieval	Legal Queries	Relevant Case Laws	Foreign
Askari and Verberne Askari and Verberne (2021)		Case Law Retrieval	Full Case Documents	Potentially Relevant Prior Cases	Foreign
Nguyen et al. Nguyen et al. (2022)		Statute Document Retrieval	Legal Queries	Relevant Statute Law Documents	Foreign
Klein et al. Klein et al. (2006)		Case Law Retrieval	Legal Key-words	Relevant Case Laws	Foreign
Maxwell and Schafer Maxwell and Schafer (2008)		Legal Information Retrieval	Legal Queries	Relevant Legal Documents	Foreign
DAGostini Bueno et al. DAGostini Bueno et al. (1999)		Intelligent Retrieval of Jurisprudential Texts	Legal Queries	Relevant Jurisprudential Texts	Foreign
van Opijnen and Santos van Opijnen and Santos (2017)		Legal Information Retrieval (IR) and Entailment	Legal Documents and Queries	Determined Whether a Query Implies Another Statement or Is Irrelevant	Foreign
Kim et al. Kim et al. (2019)		Legal IR and Entailment	Legal Documents and Queries	Determined Whether a Query Implies Another Statement or Is Irrelevant	Foreign
Mandal et al. Mandal et al. (2021)		Legal Document Similarity	Legal Document	Similar Legal Documents	Indian
Leburu-Dingalo et al. Leburu-Dingalo et al. (2020)		Precedent and Statute Retrieval	Legal Descriptions of Current Situations	Relevant Statutes and Prior Cases	Indian
Bhattacharya et al. Bhattacharya et al. (2019)		Precedent and Statute Retrieval for Indian Legal Domain	Legal Facts	Relevant Precedents and Statutes	Indian
Sampath and Durairaj Sampath and Durairaj (2022)		Precedence Retrieval from Legal Documents	Current Case Documents and Legal Domain	Similar Prior Case Documents	Indian
Pandian and Joshi Pandian and Joshi (2022)		Case Law and Statute Retrieval	Legal Queries	Relevant Laws and Statutes	Indian

Turning to LLMs in the legal domain specifically, Vats et al. (2023) [Vats et al. \(2023\)](#) used LLMs for statute prediction and judgment prediction on Indian legal judgments. They report reasonable performance on statute prediction, but judgment prediction proved harder, with gender and religion biases appearing in the model’s outputs.

Savelka et al. (2023) [Savelka et al. \(2023\)](#) explored an augmented LLM for explaining legal concepts; the unaugmented system produced noticeably less coherent explanations than the augmented one.

Yu et al. (2023) [Yu et al. \(2023\)](#) examined LLMs for legal entailment, asking whether a retrieved article entails a given query. They found that adding a legal reasoning prompt structured around Issue, Rule, Application, and Conclusion (IRAC), alongside the query and candidate articles, improved entailment performance.

Outside the Indian setting, Zheng et al. (2025) [Zheng et al. \(2025\)](#) build two legal RAG benchmarks, Bar Exam QA and Housing Statute QA, where the query and the relevant document share little lexical overlap and finding the connection between them requires legal reasoning rather than keyword matching. They prompt an LLM to generate the legal issue and applicable rule implied by the query facts, then append this to the original query before retrieval, and find that this consistently improves retrieval over the unexpanded query. It differs from this paper in three ways: the generated text expands a single query rather than being retrieved against in parallel with facts; the task is statute and opinion-passage retrieval for question answering rather than case-to-case precedent retrieval; and no reranking or explanation step follows retrieval. Separately, Santosh et al. (2025) [Santosh et al. \(2025\)](#) propose LeCoPCR, which augments a fact-based query for prior case retrieval at the European Court of Human Rights with legal concepts mined from the reasoning sections of cases using a trained model, rather than generated by an LLM from the query itself. This is further independent support for the idea that a facts-only query misses relevant grounds for retrieval, reached through a different, non-generative method in a different jurisdiction.

Two recent works extend this line of research at considerably larger scale. Paul et al. (2025) [Paul et al. \(2025\)](#) introduce IL-PCSR, an Indian Supreme Court and High Court corpus pairing 6,271 queries with a pool of 3,183 precedents and 936 statutes, modelling statute retrieval and precedent retrieval jointly; their best pipeline combines a lexical-semantic ensemble retriever with an LLM-based reranker that conditions each task’s output on the other. Han et al. (2026) [Han et al. \(2026\)](#) evaluate LLM-based citation prediction in the Australian jurisdiction on a benchmark of 55,000 instances, and find that stand-alone generative models, general-purpose or law-specific, perform poorly without retrieval or fine-tuning; instruction tuning and hybrid retrieval-reranking strategies are needed to close the gap.

This work departs from prior Indian case-law retrieval systems in three respects, though related ideas have been explored elsewhere, and in one case before the preliminary version of this work was made available. First, none of the Indian precedent retrieval systems surveyed in Table 1 pair a retrieved case with an explanation of why it is relevant. Explanation-paired legal case retrieval has been explored outside India, for instance through extracted rationales [Yu et al. \(2022\)](#) or logic rules [Sun et al.](#)

(2024), but not, to our knowledge, for Indian precedent retrieval prior to a preliminary version of this system reported in an earlier preprint; the present paper extends that system with a systematic evaluation.

Second, existing Indian case-law retrieval systems query the corpus using the facts of the current case, or in some cases full prior judgments, rather than the legal issue itself, and none combine an LLM-generated issue formulation with the factual signal in parallel. This work generates legal issue phrases directly from the query facts using an LLM, and retrieves against the facts, the issues, and their combination together rather than using one in place of the other. This follows the same idea reflected in Nigam et al.’s rhetorical-role findings Nigam et al. (2025) and Santosh et al.’s concept-augmented retrieval Santosh et al. (2025): that lawyers identify relevant precedent through the legal issue as much as through the facts. The closest prior use of this generative mechanism is Zheng et al. Zheng et al. (2025), who also use an LLM to derive issue and rule text from facts. The present paper differs by applying this to case-to-case precedent retrieval rather than question answering, by retrieving against facts and issues as separate parallel channels rather than one expanded query, and by showing that an issue-based query only helps once its length is matched to the density of the document it is retrieved against, a finding not examined in any of the work above.

Third, the pipeline uses LLMs without any task-specific fine-tuning. This fits a broader pattern in legal NLP, where zero-shot and few-shot LLM prompting has performed competitively against fine-tuned baselines on related legal tasks, though combining this with issue generation, retrieval, and reranking has not previously been evaluated for Indian case law.

The paper also sets out a concrete way to manage LLM context length limits and reduce output variability, both necessary for the approach to work in practice rather than only on paper.

3 Dataset and Corpus Construction

The FIRE 2019 AILA track Bhattacharya et al. (2019) serves as the primary benchmark for Indian legal precedent retrieval. The dataset comprises approximately 2,914 Supreme Court of India judgments and 50 queries, each representing a factual scenario for which relevant precedents must be identified. Judgments are in plain text format, ranging from 1 KB to 230 KB.

3.0.1 Need for Structured Summary Generation

Let $j \in \mathcal{J}$ denote a raw legal judgment of average length $|j| \approx 20,000$ words. Direct LLM-based retrieval requires fitting multiple judgments simultaneously into a context window of size L tokens. Since $|j| \gg L/|C|$ for most LLMs, direct retrieval over a judgment pool is practically infeasible without compression.

Figure 2 illustrates three strategies for fitting judgments into an LLM context window. Structured summaries (left) represent each judgment through key fields, allowing many cases to fit simultaneously. Paragraph-level summarisation (middle) accommodates more content than full text but less efficiently. Direct full-text input (right) fits

$$\hat{j} = \text{SlotFill}_{\theta}(j, \mathcal{T}) \quad (1)$$

where $\mathcal{T} = \{t_1, t_2, \dots, t_M\}$ is the set of M predefined template fields and SlotFill_{θ} is the LLM based extraction function. In practice, the model is asked to "Summarize the given judgment by extracting the following fields exactly": the court, the case number, the kind of judgment (appeal, petition, writ, revision, and so on), the parties involved, the date, the bench of judges, the facts, the arguments, the issues or questions framed by the court, the court's reasoning on each issue, which party the case was decided in favour of and why, the final judgment and directions, the statutes referred to, the precedents referred to, the new legal principle established, and the important subjects discussed. The model is told to return only a single JSON object, with every field filled in, using "Not mentioned" wherever a field does not apply, so that no part of the summary is ever left blank. (Figure 3).

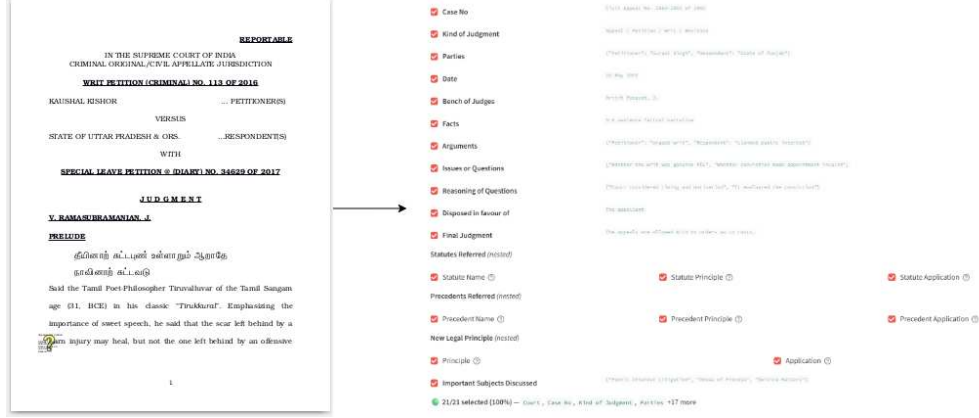


Fig. 3 (A) An unstructured Indian Supreme Court judgment (left) and (B) the 21-field structured summary template used for case law retrieval (right). Panel B shows the complete field schema.

3.0.2 Phase 1: Structured Summary Generation with Gemini Pro

In Phase 1, the template was applied to 50 judgments from the FIRE 2019 AILA corpus using Gemini Pro (temperature=0.9). The raw judgment j was segmented into overlapping text chunks using a recursive character splitter, transformed into embeddings using Google Generative AI Embedding, and stored in a FAISS vector store. Upon receiving the summarisation prompt, the most relevant chunks were retrieved via similarity search and integrated into the LLM context window, enabling generation of a structured summary $\hat{j}$ in JSON format (Figure 4).

Chunking parameters were tuned on five long judgments (Table 2; case names are abbreviated in the table to fit the page margin, with the full citation given in the table note). A recursive character splitter was chosen because Indian judgments are structured as paragraphs. Chunk sizes of 1,000 to 30,000 characters were tested

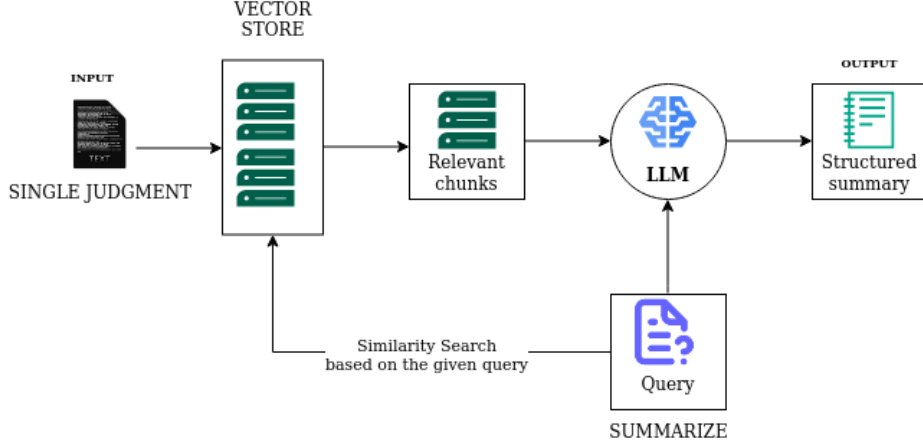


Fig. 4 Overview of the structured summary generation framework.

with overlap values of 10%–90%. A chunk size of 20,000 characters with 50% overlap and $k=4$ retrieved chunks yielded the best results: four chunks of approximately 5,000 tokens each fill the Gemini Pro context window of 30,720 input tokens. FAISS outperformed Chroma as a retriever.

Why RAG fails on long documents, and why Chain-of-Thought helps.

Complex queries require the LLM to aggregate information from multiple non-contiguous parts of the judgment. When a document exceeds approximately 122,880 characters, the answer is distributed across more than four chunks, meaning relevant content is never fully co-present in a single context window. Under simple prompting, the LLM then produces answers that are locally coherent but miss globally important content, reflected in quality ratings dropping to 2 (Table 3).

Chain-of-Thought (CoT) prompting [Wei et al. \(2022\)](#) is a technique in which the model is asked to generate explicit, step-by-step reasoning before producing its final answer, rather than producing the answer directly. Instead of mapping the input straight to an output in one inferential step, the model first writes out a chain of intermediate reasoning steps that lead to the conclusion, which has been shown to improve performance on tasks requiring multi-step reasoning.

Chain-of-Thought (CoT) prompting addresses this by instructing the model to reason step-by-step through each chunk before producing the final structured summary. This forces the model to maintain an explicit intermediate representation of partial information, integrating evidence progressively rather than relying on a single pass. With CoT, quality ratings recover to 5 across all document lengths (Table 3).

Gemini Pro’s content safety filters declined to process judgments containing sensitive legal content. This mainly affected cases involving sexual offences and violent

Table 2 Long judgments used for RAG parameter tuning.

Judgment	Words	Characters
CA 10941–10942/2013 ^a	9,968	56,435
CA 1021–1026/2013 ^b	18,292	101,107
TC (Civil) 98/2012 ^c	37,666	230,848
WP (CrI.) 113/2016 ^d	70,524	417,550
SLP (C) 9036–9038/2016 ^e	108,263	619,812

^aCivil Appeal No. 10941–10942 of 2013. ^bCivil Appeal Nos. 1021–1026 of 2013. ^cTransferred Case (Civil) No. 98 of 2012. ^dWrit Petition (Criminal) No. 113 of 2016. ^eSLP (C) Nos. 9036–9038 of 2016.

Table 3 RAG quality ratings (1–5) for long documents under three query strategies, rated by legal experts.

Judgment	Simple query	Complex query	CoT prompting
CA 10941–10942/2013	5	5	5
CA 1021–1026/2013	5	4	5
TC (Civil) 98/2012	5	2	5
WP (CrI.) 113/2016	5	2	5
SLP (C) 9036–9038/2016	5	2	5

Case labels as in Table 2.

crimes, which make up a significant portion of Indian criminal case law. As a result, the corpus was left incomplete, and retrieval evaluation had to be limited to 14 queries.

Validation of Phase 1 summaries.

As no gold-standard structured summary dataset exists for Indian legal judgments, the 50 Phase 1 summaries were validated through expert human review rather than automated checks. Fifteen randomly selected summaries were independently reviewed by three legal research scholars and two law students using a precision/recall framework via a structured assessment form (Table 4). Precision was high, at approximately 0.90 across all fields. Recall, however, was notably lower for the Statutes Referred and Precedents Referred fields, since Indian judgments routinely cite tangentially connected statutes and precedents, and the model extracted only those directly relevant to the identified legal issues. Experts confirmed this selective extraction was appropriate for a retrieval application, since exhaustive citation enumeration adds noise without improving retrieval quality, a finding later confirmed by the citation noise ablation in Phase 2 (Section 5.3.5). Experts also noted the automatically generated Issues or Questions and New Legal Principle fields as particularly useful, since these are the fields legal professionals consult when assessing a precedent’s relevance.

A second, structural problem.

Besides the recall gap, checking the summaries by hand turned up another problem with the generation step itself. The LLM did not always stick to the 21 field template.

Table 4 Expert evaluation of LLM-generated structured summaries (15 randomly selected summaries; reviewed independently by 3 legal research scholars and 2 law students via structured assessment form). Precision ≈ 0.90 across all fields; lower recall for citation fields reflects intentional selective extraction of directly relevant statutes and precedents rather than exhaustive enumeration. Recall scale: Low < 0.40 , Moderate = $0.40\text{--}0.79$, High ≥ 0.80 .

Field	Precision	Recall (approx.)	Notes
Header fields	≈ 1.00	High	Directly stated in header
Facts	≈ 0.90	Low	Key sentences selected
Arguments	≈ 0.90	Moderate	Petitioner/respondent split captured
Issues or Questions	≈ 0.90	Moderate	Primary issues captured
Reasoning of the Questions	≈ 0.90	Moderate	Follows identified issues
Final Judgment	≈ 0.90	High	Consistently extracted
Statutes Referred	≈ 0.90	Low	Only directly relevant statutes
Precedents Referred	≈ 0.90	Low	Same pattern as Statutes
New Legal Principle	≈ 0.90	Moderate	Legal principles captured
Important Subjects	≈ 0.90	Moderate	Topic tags generally accurate

Low recall for citation fields is intentional: exhaustive citation enumeration adds retrieval noise without improving precision (confirmed by the citation noise ablation, Section 5.3.5).

When the prompt did not force the model to follow the exact field boundaries, it would sometimes add its own extra sub fields, or write a field in a different shape the next time around, even for the same judgment. This got worse because of sampling temperature. Without forcing the model to give close to the same output each time, the same judgment could produce two summaries that did not even share the same structure.

Phase 1 relied on a person checking each summary by hand to catch this. That does not scale well, because every summary needed not just a check for accuracy, but also a check that it followed the agreed format at all. This led to two changes for Phase 2: using a bigger corpus, and building a generation pipeline that enforces the template by design, instead of relying on the prompt and manual checking alone.

This problem, along with the content safety filter issue mentioned earlier, is why the project moved to Llama 3 70B for Phase 2.

3.0.3 Phase 2: Structured Summary Generation with Llama 3 70B

Phase 2 uses `llama-3-3-70b-instruct` to generate summaries for 500 documents from the FIRE 2019 AILA corpus, the largest working corpus the available API budget and time constraints allowed. The slot-filling framing and 21-field schema remain consistent with Phase 1; the key differences are the LLM, greedy decoding (temperature = 0, seed 42), and a validation and repair pipeline designed specifically to address the structural drift identified in Phase 1.

Phase 1 used one fixed way of reading every judgment: split into chunks of 20,000 characters with 50 percent overlap, retrieve the top 4 chunks, and answer one combined question in a single LLM call. This worked for 50 judgments from the FIRE corpus that a person could check afterwards, but it could not scale to the 496 judgments, where document length varies enormously and statute or precedent recall was already weak at around 60 percent. Phase 2 fixes this by first measuring each file’s size, word count, and page count, then routing short judgments through a single pass and long judgments through four separate passes, each asking for only a handful of fields with its own retrieval query built from natural judgment vocabulary rather than the instruction text itself. The statutes and precedents pass also retrieves more chunks than the others, since citations tend to be scattered throughout a judgment rather than grouped together. The model’s temperature and seed are fixed, and retrieved chunks are sorted by content hash before being shown to the model, so the same document produces the same summary every time it is run, which Phase 1 never guaranteed.

The second set of changes deals with what happens once the LLM responds, since Generation 2 showed that prompting alone is not reliable at this scale. Phase 2 reshapes fields like Parties, Arguments, and the New Legal Principle into their correct structure if the model returns them in the wrong shape, and its JSON parser is built to recover a usable object even from malformed or truncated output. Any field still missing afterwards goes through a first retry using a keyword based search, and then a second, more aggressive retry that tries multiple phrasings field by field before falling back to raw extracted sentences as a last resort. A separate quality check flags statute and precedent entries whose explanations are present but too thin to be useful, something a human reviewer would have caught by eye in Phase 1 but which has to be written into the code at 496 documents.

Of 500 judgments processed, 496 passed all validation tiers. In the final corpus, all four required fields (Court, Parties, Facts, Issues) achieve 100% coverage; Reasoning of the Questions reaches 97.8%, Statutes Referred 96.4%, and Precedents Referred 95.6%. The judgments (C71, C342, C406, C481) failed due to a combination of document length, which produced irreconcilable multi-pass merge conflicts, and transient infrastructure faults during generation. C71 appears as a ground-truth case for queries Q28(C71,C178) and Q37(C18,C71,C178), imposing an unrecoverable ceiling deficit on those queries in all structured-summary experiments (see Section 5.3.9).

Within the budget and time constraints noted above, 500 documents was the largest slice we could process; we chose the first 500 specifically because it is the smallest such slice of the 2,914-document FIRE 2019 AILA corpus that still contains all relevant documents for every one of the 50 queries, which also means the corpus can be recreated exactly by another researcher without needing our random seed. One query was blocked by Gemini’s safety filter during Phase 1 (Q46), and the four documents listed above failed during Phase 2 summary generation, slightly reducing recall for queries Q28 and Q37 as noted.

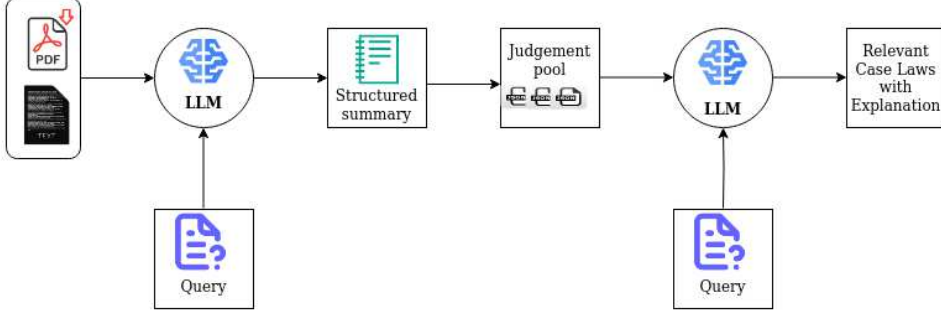


Fig. 5 Schematic overview of the proposed AQgR framework. The proposed framework operates in two stages: structured summary generation, followed by ensemble retrieval with LLM-based selection. Two distinct queries drive the pipeline — one for structured summarization, supplied either as a user query or an LLM prompt, and one for case-law retrieval, which can be the facts of the case, the legal issues, or any other search criterion.

4 Methodology

This section describes the AQgR framework for Indian Supreme Court precedent retrieval. The system takes two inputs: the corpus of structured judgment summaries described in Section 3, and a query consisting of the facts of a current case. The output is a ranked list of relevant prior judgments with explanation of why each was selected. Figure 5 shows the overall framework. The proposed framework operates in two stages. In the first stage, structured summaries are generated from the source judgments. These structured summaries are then passed to the second stage, where an ensemble retrieval strategy is applied, followed by LLM-based selection to identify the most relevant cases.

4.1 Problem Formulation

Let $\mathcal{J} = \{j_1, j_2, \dots, j_N\}$ denote the corpus of N structured judgment summaries and let q denote a query consisting of the factual scenario of a current case. The goal is to produce a ranked list $\mathcal{R} \subseteq \mathcal{J}$ of prior judgments whose legal principles are relevant to the legal issues raised by q , regardless of factual similarity. We denote the ground-truth relevant set as $\mathcal{G}(q) \subseteq \mathcal{J}$.

The key challenge is that legal relevance is determined by shared doctrine, not shared facts. Two cases with entirely unrelated facts may invoke the same legal principle — a bank officer’s termination and an electricity board worker’s termination are both governed by Article 14 of the Constitution, yet share no factual vocabulary. The retrieval function $f : q \rightarrow \mathcal{R}$ must therefore rank judgments by legal doctrine alignment rather than surface-level factual similarity. The AQgR framework decomposes f into three sub-functions, illustrated in detail in Figure 6:

$$\mathcal{I}(q) = \text{LLM}_\theta(q) = \{i_1, i_2, i_3\} \quad (2)$$

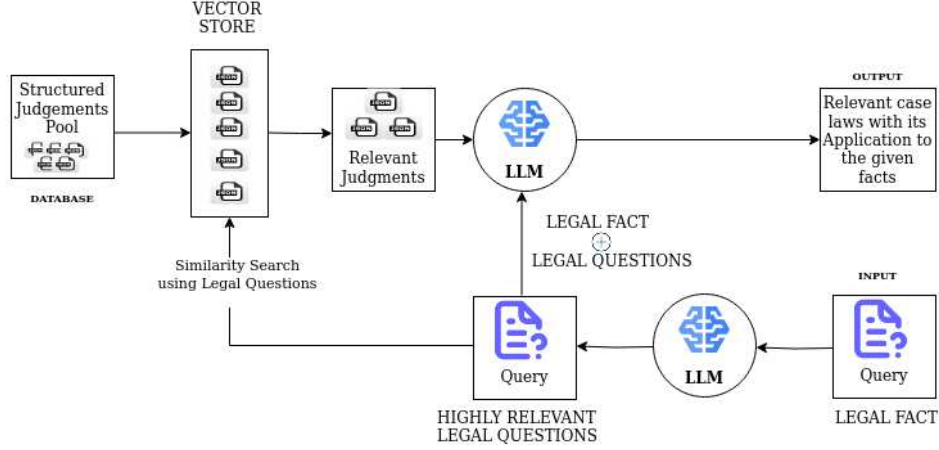


Fig. 6 Detailed pipeline for case law retrieval (schematic). Left: structured judgment summaries are stored in a vector store (FAISS) and a BM25 index. Centre: the issues query q_I , generated from the query facts by the issue-generation LLM, is used for first-stage B5 retrieval; the resulting pool is carried forward as the candidate pool for reranking. Right: the reranking LLM receives the facts+issues query q_{FI} together with the candidate pool and returns relevant case laws with per-case explanations. LEGAL FACT = facts of the current query case; LEGAL QUESTIONS = LLM-generated legal issue phrases (q_I);

where $\mathcal{I}(q)$ is the set of three abstract legal issue phrases, 8–15 words each, generated from the factual query q .

$$\mathcal{C}(q_I) = \text{Retrieve}(q_I), \quad \text{where } q_I = i_1 \mid i_2 \mid i_3 \quad (3)$$

where $\mathcal{C}(q_I)$ is the candidate pool returned by first-stage retrieval on the issues query q_I , and $\text{Retrieve}(\cdot)$ denotes the proposed B5 hybrid retriever (BM25 weighted 0.7, FAISS weighted 0.3; Equation 6). q_F and q_{FI} are also retrieved in parallel for comparison (Section 5.3.2), but q_I alone is carried forward into reranking: pooling all three by union dilutes the ranking the LLM reranker receives, lowering its result even though the pool’s recall ceiling improves (Section 5.3.6).

$$\mathcal{R}(q) = \text{LLM}_\theta(\mathcal{C}(q_I), q_{FI}) \quad (4)$$

where $\mathcal{R}(q)$ is the final ranked output selected by the LLM from $\mathcal{C}(q_I)$ using the facts+issues query q_{FI} as the reranking signal. The short issue query q_I achieves its best result only under document format V4, the compressed structured summary that states the legal principle behind each citation (Experiment 5, Section 5.3.5); under the other three formats, a longer or more informative query is needed for the LLM to reason well at reranking. The practical implication follows directly: if the document format is fixed to V4, a short issue query is sufficient and is the recommended choice for retrieval. If the document format is not fixed, a more informative query works better with the LLM reranker than a short query alone.

4.2 Legal Issue Generation

Given the facts of a query case, the system makes a single LLM call to compute $\mathcal{I}(q)$. The LLM generates three abstract legal issue phrases formulated in the vocabulary of Indian Supreme Court legal doctrine, naming the constitutional provision, legal doctrine, or procedural principle at stake rather than the facts of the case. The prompt instructs the model to avoid proper nouns and keep phrases abstract enough to match prior cases with entirely different factual circumstances.

In Phase 1, we used simple legal issue generation without any length constraint, resembling the long issues tested in the Phase 2 experiments. However, we found that using long LLM-generated queries for retrieval makes the output non-deterministic. For real-world applications, optimum-length queries are preferable, but for this specific dataset and for reproducibility, short-length queries proved better.

For example, given facts about an employee’s termination, the model generates “*arbitrary termination without inquiry violating Article 14 natural justice*” rather than describing the specific employer or employee. This abstraction enables cross-domain retrieval: the phrase matches prior cases about termination in any employment context. The three generated phrases are concatenated into a single pipe-separated string to form q_I :

$$q_I = i_1 \mid i_2 \mid i_3 \quad (5)$$

Concatenating the three phrases into a single query, rather than running three separate queries, is a deliberate design choice driven by the density alignment principle: a combined query provides enough lexical density to match effectively against structured summaries through BM25 and FAISS, whereas individual short phrases of 8–15 words cannot supply enough IDF-weighted signal on their own.

4.3 Query Formulations

Three query formulations are derived from q and $\mathcal{I}(q)$:

- q_F : the raw factual narrative of the current case.
- q_I : the three issue phrases concatenated with pipe separators, as defined in Equation 5.
- q_{FI} : the factual narrative concatenated with the issue phrases, combining both signals.

All three are matched against the structured summary corpus at retrieval to compare performance, but each plays a different role in the final pipeline, because retrieval and reranking are not sensitive to the same things. B5 retrieval is a similarity-matching method built on BM25 and FAISS, so it depends on how closely the query’s lexical density matches the document’s: q_I is short and concentrated in doctrine terms, which matches the structured summary’s density best and gives the strongest direct retrieval result, while q_F is denser than the structured summary calls for and underperforms at this stage on its own. The LLM reranker has no such density constraint, since it reads and reasons over the full text it is given, so a longer, narrative query does not work against it the way it does against BM25 and FAISS. This is why q_F helps at reranking despite not being the best retrieval query: the LLM can use the full factual

narrative to reason about doctrinal applicability, something short issue phrases alone do not give it enough context to do. q_{FI} keeps the doctrinal anchor of q_I while adding the narrative context of q_F , which is why it is used as the reranking query rather than q_F or q_I alone.

4.4 First-Stage Retrieval

First-stage retrieval narrows the full corpus down to a candidate pool of 100 documents using a hybrid lexical-semantic similarity search, denoted B5. Combining lexical and semantic retrieval signals is itself an established technique in this literature rather than a contribution of this paper: IL-PCSR’s best pipeline also uses a lexical-semantic ensemble retriever [Paul et al. \(2025\)](#), and TraceRetriever fuses BM25, dense vector retrieval, and a cross-encoder via Reciprocal Rank Fusion [Nigam et al. \(2025\)](#). What this paper adds at the retrieval stage is not the hybrid mechanism itself but the density-alignment behaviour: the same B5 hybrid favours short or long, fact- or issue-based queries depending on how closely the query’s lexical density matches the document representation it is scored against, a query-document interaction not examined under either prior hybrid pipeline. For a query q_I , each candidate document $j \in \mathcal{J}$ is scored by combining a lexical score from BM25 and a semantic score from FAISS dense retrieval:

$$\text{score}(q_I, j) = 0.7 \cdot \widehat{\text{BM25}}(q_I, j) + 0.3 \cdot \widehat{\text{FAISS}}(q_I, j) \quad (6)$$

where $\widehat{\text{BM25}}$ and $\widehat{\text{FAISS}}$ are the BM25 and FAISS similarity scores, each min-max normalised to $[0, 1]$ before combination. The top 100 documents by this combined score form the candidate pool $\mathcal{C}(q_I)$ carried forward to the LLM reranking stage (Section 4.5).

We tested five BM25/FAISS weighting combinations, ranging from pure FAISS (0/1) to pure BM25 (1/0). BM25 contributed more retrieval signal than FAISS on its own: pure BM25 outperformed pure FAISS on every metric at both the bare-retrieval stage (MAP 0.1619 versus 0.0831, alongside higher P@10 and MRR) and after LLM reranking (MAP 0.1616 versus 0.1129, with the same pattern on the remaining metrics). FAISS’s contribution is, however, uneven rather than uniformly harmful: at bare retrieval, MAP rises monotonically with the BM25 weight and pure BM25 wins outright, but after LLM reranking the 0.3/0.7 split overtakes every other weighting on MAP and MRR, including pure BM25. This indicates that FAISS and BM25 do not consistently recover the same documents — FAISS surfaces relevant cases that BM25 misses in some queries, though not reliably enough to beat BM25 on its own — so a purely lexical retriever would forgo coverage that a hybrid recovers in at least some cases, even though that benefit does not show up at every weighting or every stage. We adopted the 0.7/0.3 BM25/FAISS split in Equation 6 as the default, keeping BM25 as the dominant signal while retaining a FAISS contribution large enough to capture some of this benefit without taking on the larger, less stable gains (and corresponding risk) seen at higher FAISS weightings.

For the FAISS semantic component, we use all-MiniLM embeddings (Sentence-BERT), chosen for their speed. We also tried Legal-BERT embeddings, but these gave noticeably worse results — likely because Legal-BERT is trained for legal language modelling rather than for similarity search specifically, whereas all-MiniLM is trained

Table 5 BM25/FAISS weighting sweep, bare retrieval and after LLM reranking.

BM25 / FAISS	MAP	P@10	MRR
<i>Bare retrieval</i>			
0.0 / 1.0	0.0831	0.0620	0.1780
0.3 / 0.7	0.0989	0.0620	0.2078
0.5 / 0.5	0.1306	0.0900	0.2867
0.7 / 0.3	0.1382	0.0940	0.2903
1.0 / 0.0	0.1619	0.0918	0.3141
<i>After LLM reranking</i>			
0.0 / 1.0	0.1129	0.0820	0.1959
0.3 / 0.7	0.1689	0.0900	0.3158
0.5 / 0.5	0.1418	0.0800	0.2821
0.7 / 0.3	0.1508	0.1020	0.2945
1.0 / 0.0	0.1616	0.0918	0.3231

with a sentence-similarity objective on pairs of similar and contradictory sentences, which aligns directly with the retrieval task at hand.

4.5 LLM Reranking

The candidate pool $\mathcal{C}(q_I)$ returned by the issues query q_I is passed to the LLM in a single call. Each candidate document is represented using a configurable subset of fields from its structured summary, chosen by the user (for example, legal principle, court, parties, date, facts, issues or questions, reasoning, final judgment, new legal principle, and important subjects), with the full content of the selected fields passed directly to the LLM. As an alternative, a fixed 600-character extraction following a set field order can also be used. However, the former approach, allowing user-selected fields with their full content, proved to be the more effective of the two. The LLM is prompted to select and rank the top-20 most legally relevant cases, returning one case identifier and a justification per selection.

As stated in Equation 4, the facts+issues query q_{FI} is used as the reranking signal, not q_F or q_I alone. q_I gives the best direct retrieval result, but short issue phrases alone do not give the LLM enough to reason about doctrinal alignment. q_F gives the LLM a full narrative to reason over, but it is too dense for the retrieval stage’s structured summary. q_{FI} supplies both: the issue phrases anchor the doctrine, and the facts give the LLM the context to reason about it, which is why it is the strongest reranking signal of the three. Here we used a single prompt. In practice, however, it is better to keep multiple prompts, since the prompt that works best can vary depending on the format of the input documents.

5 Experiments and Results

5.1 Evaluation Protocol

Evaluation software and metric conventions. We compute all IR metrics ourselves in Python, using standard definitions for average precision, precision/recall at

k , NDCG, and reciprocal rank. Relevance is binary: a document is either relevant (1) or not (0). NDCG uses the standard $1/\log_2(k+1)$ discount. We do not run significance tests except for Experiment 8, since all comparisons are paired runs on the same 50 queries; instead we report effect sizes and metric values directly to support our conclusions.

How a method is scored. For every query, each method produces its own ranked output list — this could be a retrieval ranking, an LLM’s selected and ordered shortlist, or a fused ranking from multiple prompts. We score that list directly: we walk down it in order and check, at each position, whether the document there is one of the true relevant documents for that query. We never look beyond what the method itself returned: if a relevant document is missing from a method’s output, it is simply treated as not found, with no credit added back from outside that list.

Formally, for query q let $\mathcal{R}(q)$ be the ranked list a method returns and $\mathcal{G}(q)$ the ground-truth relevant set. We evaluate against

$$\mathcal{P}(q) = \mathcal{R}(q) \cup \mathcal{G}(q) \quad (7)$$

which simply means that relevance is judged using $\mathcal{R}(q)$ ’s own ranking, and any ground-truth document the method did not return at all is scored as a miss rather than left unjudged. The union does not give the method credit for documents it never returned — it only guarantees that missed ground-truth documents still count against the score, rather than being quietly excluded from it.

Table 6 A running example used throughout this subsection. Three documents are ground truth, $\mathcal{G}(q) = \{C_9, C_{14}, C_{22}\}$. A method returns five ranked documents; C_{22} is never returned.

Rank k	Document	Relevant?	rel(k)
1	C_9	yes	1
2	C_{40}	no	0
3	C_{14}	yes	1
4	C_{82}	no	0
5	C_{90}	no	0
—	C_{22} (never returned)	yes	—

A running example. Table 6 defines one running example that we carry through every metric below, so that each formula’s output can be checked by hand. The rank- k relevance indicator is $\text{rel}(k) = 1$ if the document at rank k is in $\mathcal{G}(q)$ and 0 otherwise, giving the sequence $\text{rel} = (1, 0, 1, 0, 0)$ for this list.

Phase 1 evaluation. Phase 1 follows the FIRE 2019 evaluation protocol [Bhattacharya et al. \(2019\)](#), in which ground-truth relevance judgements carry no ranking information, so retrieval is scored as rank-unaware:

$$\text{Precision}(q) = \frac{N_{\text{rel}}(q)}{N_{\text{ret}}(q)}, \quad \text{Recall}(q) = \frac{N_{\text{rel}}(q)}{|\mathcal{G}(q)|} \quad (8)$$

On the running example, 2 of the 5 returned documents are relevant, so $\text{Precision}(q) = 2/5 = 0.4$, and against the 3 true ground-truth documents $\text{Recall}(q) = 2/3 \approx 0.667$. Since ranking is not considered, $\text{AP}(q)$ reduces to per-query $\text{Precision}(q)$; MAP and MAR (Mean Average Recall) average these over queries. One query (Q46) was blocked by Gemini Pro’s content safety filter and is excluded from the denominator, giving $|Q| = 14$.

Phase 2 evaluation. Phase 2 uses standard ranking-aware IR metrics throughout, since retrieval, cross-encoder filtering, single-prompt re-ranking, and RRF all produce an explicitly ranked output list. For query q , let $d_1, d_2, \dots, d_n$ denote the ranked list defined by Eq. 7, and let $\text{rel}(k) = 1$ if $d_k \in \mathcal{G}(q)$ and 0 otherwise, as in Table 6.

Average Precision. The running precision at each relevant rank, summed and normalised by the size of the ground-truth set:

$$\text{AP}(q) = \frac{\sum_{k=1}^n P@k \cdot \text{rel}(k)}{|\mathcal{G}(q)|}, \quad P@k = \frac{1}{k} \sum_{j=1}^k \text{rel}(j) \quad (9)$$

with $\text{AP}(q) = 0$ when $|\mathcal{G}(q)| = 0$. On the running example, $P@1=1.0$, $P@2=0.5$, $P@3 \approx 0.667$, $P@4=0.5$, $P@5=0.4$; only ranks 1 and 3 contribute a term, so

$$\text{AP}(q) = \frac{(1.0)(1) + (0.5)(0) + (0.667)(1) + (0.5)(0) + (0.4)(0)}{3} \approx 0.556 \quad (10)$$

The missed document C_{22} never contributes to the numerator: it only enlarges the denominator, pulling the score down from what two perfectly-ranked hits alone would give.

Precision and recall at cutoff K . Computed over the first K documents of the ranked list only:

$$P@K(q) = \frac{1}{K} \sum_{k=1}^K \text{rel}(k), \quad R@K(q) = \frac{1}{|\mathcal{G}(q)|} \sum_{k=1}^K \text{rel}(k) \quad (11)$$

with $R@K(q) = 0$ when $|\mathcal{G}(q)| = 0$. At $K = 5$ on the running example: $P@5 = 2/5 = 0.4$ and $R@5 = 2/3 \approx 0.667$.

NDCG@ K . The ratio of achieved to ideal discounted cumulative gain:

$$\text{NDCG}@K(q) = \frac{\text{DCG}@K(q)}{\text{IDCG}@K(q)}, \quad \text{DCG}@K(q) = \sum_{k=1}^K \frac{\text{rel}(k)}{\log_2(k+1)} \quad (12)$$

$$\text{IDCG}@K(q) = \sum_{i=0}^{\min(|\mathcal{G}(q)|, K)-1} \frac{1}{\log_2(i+2)} \quad (13)$$

with $\text{NDCG}@K(q) = 0$ when $\text{IDCG}@K(q) = 0$. On the running example at $K = 5$, the two hits at ranks 1 and 3 give $\text{DCG}@5 = 1/\log_2 2 + 1/\log_2 4 = 1.0 + 0.5 = 1.5$. The ideal ranking places all 3 ground-truth documents at ranks 1–3: $\text{IDCG}@5 =$

$1/\log_2 2 + 1/\log_2 3 + 1/\log_2 4 \approx 2.131$. So $\text{NDCG@5} = 1.5/2.131 \approx 0.704$ — lower than AP, since NDCG additionally penalises the second hit for landing at rank 3 instead of rank 2.

MRR. The reciprocal of the rank of the first relevant document found, used throughout Phase 2 as the cutoff-independent comparator whenever pool definitions or selection counts differ across runs (e.g. retrieval-only versus LLM re-ranked, issues-only pool versus union pool):

$$\text{MRR}(q) = \frac{1}{\min\{k : \text{rel}(k) = 1\}} \quad (14)$$

with $\text{MRR}(q) = 0$ if no relevant document is retrieved. On the running example, the first hit (C_9) is at rank 1, so $\text{MRR}(q) = 1.0$. MRR only credits where the *first* hit lands, so it is unaffected by C_{14} ’s later rank or by C_{22} being missed entirely — which is exactly why it remains comparable across runs with different list lengths or pool sizes, unlike AP, P@K, R@K, or NDCG.

Per-query metrics are averaged across all queries in Q to obtain the reported run-level scores, following the same pattern as MAP:

$$\text{MAP} = \frac{1}{|Q|} \sum_{q \in Q} \text{AP}(q) \quad (15)$$

and identically for P@10, P@20, R@20, NDCG@10, NDCG@20, and MRR; queries with zero ground-truth documents are excluded from the relevant averages where the denominator would be undefined.

Table 7 Every metric evaluated on the running example (Table 6, $K=5$).

Metric	Value
Precision (Phase 1, rank-unaware)	0.400
Recall (Phase 1, rank-unaware)	0.667
AP	0.556
P@5	0.400
R@5	0.667
NDCG@5	0.704
MRR	1.000

What $\mathcal{R}(q)$ is, by stage. $\mathcal{R}(q)$ is defined differently depending on which stage of the pipeline is being evaluated: it is the top 100 (for some experiments it is 150) retrieved documents at first-stage retrieval, the combined union pool for union runs (Exp. 2, 5, 6), the top-40 set after cross-encoder filtering (Exp. 4), or the LLM’s own selected list (top-20 by default, top-30 in Exp. 2/5/6, or the RRF-fused top-20 in Exp. 3/4). Because the list length and source differ across stages, MAP and other cutoff-sensitive metrics are only meaningfully compared *within* a given stage and pool definition, not across them. MRR — as shown on the running example above — is

far less sensitive to list length, so we use it as the main metric whenever we compare across stages, pool types, or selection counts.

Recall ceiling. The *recall ceiling* measures the maximum recall achievable by any downstream re-ranking stage, independent of prompt or LLM quality:

$$\text{Ceiling}(q) = \frac{|\mathcal{C}(q) \cap \mathcal{G}(q)|}{|\mathcal{G}(q)|} \quad (16)$$

where $\mathcal{C}(q)$ is the pre-rerank candidate pool — single-variant retrieval, the round-robin union pool, or the cross-encoder-filtered set, depending on the experiment — so ceiling values are only comparable within the same pool definition. On the running example, if the pre-rerank candidate pool contained C_9 and C_{14} but not C_{22} , the ceiling would be $2/3 \approx 0.667$, the same value as $R@5$ here, since all the candidates the re-ranking stage receives in this example coincide with the ranked list. Ceiling is averaged over all 50 queries, excluding queries with zero ground-truth documents.

Reciprocal Rank Fusion. For parallel-prompt runs (Exp. 3 onward), candidate rankings from multiple prompts are fused by:

$$\text{score}(c) = \sum_p \frac{1}{\text{rank}_p(c) + 60} \quad (17)$$

using the standard constant $k=60$ [Cormack et al. \(2009\)](#); a candidate not selected under a given prompt p contributes zero for that prompt. The fused ranking is truncated to the top 20 and evaluated identically to any other LLM run.

Cross-encoder filtering. Experiment 4 re-scores each retrieval pool with `cross-encoder/ms-marco-MiniLM-L-6-v2` (22M parameters) on the facts query, truncating to the top 40 documents before LLM re-ranking. This filtered set serves as $\mathcal{R}(q)$ and $\mathcal{C}(q)$ for all downstream metrics in that experiment.

Union pooling. For Experiments with union pool, the union pool is built by round-robin deduplication across query-variant retrieval runs — one document taken from each variant’s ranked list in turn, skipping duplicates — so that no single query formulation dominates the merged candidate list.

Run-to-run drift.

To test pipeline stability (Experiment 8), each configuration is run twice under identical settings, and the per-query absolute drift is measured for a metric X (e.g. AP, NDCG@20) and configuration v :

$$\Delta_v(q) = \left| X_v^{(2)}(q) - X_v^{(1)}(q) \right|, \quad q \in Q, \quad (18)$$

where $X_v^{(1)}(q)$ and $X_v^{(2)}(q)$ are the value of X for query q under run 1 and run 2. We report the mean drift $\bar{\Delta}_v = \frac{1}{|Q|} \sum_{q \in Q} \Delta_v(q)$, the fraction of queries with exact agreement ($\Delta_v(q) = 0$), and a Wilcoxon signed-rank test or Mann–Whitney U test (as appropriate) on the paired or grouped per-query values, across all eight core metrics (AP, P@10, P@20, R@20, F1@20, NDCG@10, NDCG@20, MRR).

Comparability across phases and with prior work. Phase 1 (rank-unaware, 14 queries, 50-judgment pool) and Phase 2 (ranking-aware, 50 queries, 496-judgment pool) MAP values are not comparable to each other, nor to the FIRE 2019/2020 leaderboard scores, which were computed on the full 2,914-document plain-text corpus. All Phase 2 baselines (B1–B4) are re-evaluated on the same 496-document corpus under identical conditions. Within Phase 2, MAP and other cutoff-sensitive metrics are comparable only within a matched pool definition and selection count; MRR is the metric used to compare across pool types, selection counts, and experiments.

5.2 Phase 1: Proof of Concept (Gemini Pro)

The AQgR pipeline was tested with and without the Precedents Referred field in the structured summary. Table 8 reports per-query precision and recall for both configurations.

Table 8 Phase 1 retrieval results (Gemini Pro, 50-judgment corpus, 14 queries after content-safety exclusions). One query (Q46) was declined by Gemini Pro’s content safety filter and are excluded from the denominator. MAP and MAR are rank-unaware averages of precision and recall over 14 queries.

Query	Without Precedents		With Precedents	
	Precision	Recall	Precision	Recall
Q1	0.400	1.000	0.333	1.000
Q2	0.667	1.000	0.667	1.000
Q3	0.000	0.000	0.000	0.000
Q4	0.200	1.000	0.000	0.000
Q5	0.333	0.333	0.000	0.000
Q31	0.667	1.000	0.667	1.000
Q32	0.200	0.333	0.125	0.333
Q33	0.333	1.000	0.250	1.000
Q34	0.250	0.500	0.290	0.500
Q35	0.200	0.500	0.000	0.000
Q47	0.125	1.000	0.000	0.000
Q48	1.000	0.750	1.000	0.750
Q49	0.400	0.500	0.380	0.750
Q50	0.333	0.500	0.660	1.000
Average (MAP / MAR)	0.365	0.672	0.313	0.524

MAP = Mean Average Precision (rank-unaware); MAR = Mean Average Recall. Query Q46 excluded (content safety refusals); denominator = 14 queries. These results are not comparable with FIRE 2019/2020 leaderboard scores (different corpus and protocol).

Without including precedents in the summary, the system achieves MAP 0.365 and MAR 0.672. With precedents included, MAP drops to 0.313 and MAR to 0.524. The decrease occurs because including the full Precedents Referred block expands the document representation with tangentially related cases and adds noise in the LLM context. This finding motivates the citation noise ablation in Phase 2 (Experiment 5).

Note that these Phase 1 figures are rank-unaware precision values computed over 14 queries on a 50-judgment closed pool. They cannot be directly compared with the FIRE 2019/2020 leaderboard MAP values or with Phase 2 results.

5.2.1 Qualitative Evaluation — Explanation Quality

The structured summaries were evaluated by legal research scholars through a Google Form circulated for this purpose. The relevance explanations accompanying each retrieved case were separately validated by a practising lawyer, who was given a simple formatted report showing the generated legal issues and the retrieved cases with their relevance explanations (sample in Appendix A). Feedback was positive: the lawyers found the automatic generation of legal questions helpful, and the lawyer confirmed that the explanation alongside each case allowed a reader to decide whether to read the full judgment without spending time on irrelevant results.

This evaluation has significant limitations that should be stated clearly. The structured-summary assessment was carried out by law students and research scholars, not practising lawyers, using a Google Form. The relevance-explanation assessment was carried out by a single practising lawyer, with no rubric or structured scoring criteria at all. In both cases, no inter-rater reliability check was performed. This constitutes qualitative feedback from a convenience sample, not a validated usability study. We do not use this evaluation to support any quantitative claim about explanation quality. Its purpose is to confirm that the output format is understandable to legal readers and to motivate retaining the explanation capability in Phase 2, which it does. A proper evaluation with multiple practising lawyers, a validated rubric, and inter-rater agreement measurement is left as future work.

5.2.2 Discussion

Two results from this small-scale test motivate choices made in Phase 2. First, the drop in MAP and MAR with the Precedents Referred field is a representation problem, not a retrieval one: each precedent is described with its name, court, and a short relevance statement, and only part of this speaks to the query, so the document grows longer without matching more tightly. This motivates testing document fields directly in Phase 2, and specifically comparing citation names against citation principles in Experiment 5. Second, one query out of fifteen was declined by the model’s safety filter at this small scale, a dependency on a single model’s safety behaviour that could remove more judgments as the corpus grows; Phase 2 avoids this by moving to Llama 3 70B, though it introduces a different failure mode at the summary-generation stage instead. Beyond these two points, Phase 1’s role is to show the overall approach is workable at small scale; the finer questions of query formulation and document fields are taken up in Phase 2.

5.3 Phase 2: Systematic Ablation Study (Llama 3 70B)

Phase 2 runs a series of controlled experiments that test, one variable at a time, how query formulation, document representation, retrieval pooling, and re-ranking design affect case retrieval quality. All experiments use Llama 3 70B as the re-ranking model

Table 9 Overview of the eight Phase 2 experiments. Each experiment is motivated by a finding from the one before it.

Exp.	Question	Motivated by
1	Do facts or issues retrieve and re-rank better, on plain text versus structured JSON?	Establishing a Phase 1 baseline
2	Can a union of facts, issues, and combined pools recover cases that single queries miss?	Exp. 1: low recall ceiling
3	Does the choice of re-ranking prompt matter, and does combining prompts help?	Exp. 2: improved recall but not MAP
4	Does filtering the pool with a cross-encoder before LLM re-ranking help?	Exp. 3: LLM still misses cases present in the context
5	Which structured document format, and which issue length, works best for re-ranking?	Phase 1: removing citations changed re-ranking quality, suggesting different document fields give different outputs
6	Does fusing facts and issues into one pool cost more in rank quality than it gains in coverage?	Exp. 5: failure analysis on the FI query
7	Can a short, compressed facts query replace or complement the issues query?	A new Exp. motivated by Exp. 5’s density matching finding
8	What causes the run-to-run inconsistency under identical settings, and for LLM-generated retrieval text, is short or long text more stable?	Exps. 2/3/6: identical configs gave different results on re-run

and are evaluated on the AILA 2019 FIRE benchmark of 50 queries, following the standard TREC-style MAP, MRR, $P@k$, $R@k$, and $NDCG@k$ protocol.

Part A and Part B in Experiment 1 use the full 500-judgment plain-text corpus from the FIRE dataset. All later experiments use 496 structured JSON documents, produced by converting the same corpus into multi-field summaries (four documents were excluded due to pre-processing failures). Table 9 gives a short map of all eight experiments and the motivation behind each one.

Naming convention.

Throughout this section, B1 to B5 refer to the five retrieval methods under comparison: **B1** is pure BM25 (BM25Okapi with stopwords removal, corresponding to the IITP baseline from FIRE 2019 [Bhattacharya et al. \(2019\)](#)); **B2** is TF-IDF cosine similarity with unigram and bigram features (HGC_1/Kavya system); **B3** is a weighted TF-IDF and BM25 hybrid, $0.7 \times \text{BM25} + 0.3 \times \text{TF-IDF}$ (HGC_2 system); **B4** is IDF keyword extraction, retaining the top-50% IDF-weighted tokens before running BM25 on the reduced query (HLJIT best system); and **B5** is our proposed BM25+FAISS hybrid (Equation 6).

A run label is built by stacking up to four components in order: retrieval method, retrieval-stage query type, issue-query length (where applicable), and reranking-stage query type. F, I, and FI mark the query used for retrieval — facts only, LLM-generated

issues only, or both combined — so B1_F denotes pure BM25 retrieval with a facts-only query and B5_I denotes the proposed method with an issues-only query. A *short* or *long* suffix on I specifies the issue query itself: *short* is the three 8–15 word issue keyword phrases, and *long* is the expanded 40–80 word formal legal questions, so B5_I_SHORT and B5_I_LONG denote the proposed method retrieving on the short and long issues queries respectively. B5U_SHORT and B5U_LONG apply the same distinction to the union pool. A suffix beginning with L marks the query used at the LLM reranking stage: in B5_I_LF, the B5_I portion identifies the candidate pool (the proposed B5 method with the issues query at retrieval), L marks the transition to reranking, and the letter that follows — here F — names the reranking query, so the pool is reranked using the facts query. Where both the issue-length suffix and the reranking suffix apply, they stack in the same order: B5_I_SHORT_LI denotes the proposed B5 method, retrieved on the short issues query, and reranked using the issues prompt. B5U denotes the union pool built from all three B5 query variants (facts, issues, facts combined with issues).

Table 10 Legend for run labels used throughout this section. A full label stacks components left to right in the order shown.

Position	Component	Meaning
1	B1	Pure BM25
	B2	TF-IDF cosine (unigram + bigram)
	B3	TF-IDF + BM25 hybrid (0.7/0.3)
	B4	IDF keyword extraction + BM25
	B5	Proposed BM25+FAISS hybrid
	B5U	Union pool ($B5_F \cup B5_I \cup B5_FI$)
2	_F	Retrieval query: facts only
	_I	Retrieval query: issues only
	_FI	Retrieval query: facts + issues
3	_short	Issue query: 8–15 word phrases
	_long	Issue query: 40–80 word questions
	(omit if not varied)	
4	_LX	Reranking query, $X \in \{F, I, FI\}$

Worked examples: B5_I_LF = B5 method, retrieved on issues query, reranked on facts query. B5_I_SHORT_LI = B5 method, retrieved on the *short* issues query, reranked on the issues query.

Shared settings.

Unless an experiment states otherwise, the following hold throughout Phase 2. Issue queries are short, 8 to 15 word phrases generated by the LLM. All LLM calls, for issue generation and for re-ranking alike, use temperature 0 with greedy decoding, so that differences in MAP can be attributed to the design choice under test rather than to sampling noise. Retrieval returns the top 100 documents per query, and the LLM selects the top 20 of these for its final ranking. The re-ranking prompt asks the LLM to choose the most relevant precedents for the current case, in order from most to least

Table 11 MAP across query variants and retrieval methods, plain-text corpus, short issue queries.

Query Variant	B1 BM25	B2 TF-IDF	B3 Hybrid	B4 IDF+BM25
Facts only	0.1853	0.2049	0.2040	0.1971
Issue 1 only	0.1276	0.1562	0.1479	0.1374
Issue 2 only	0.1309	0.1086	0.1310	0.1151
Issue 3 only	0.1336	0.1244	0.1437	0.1304
RRF (Facts + I1-I3)	0.1592	0.1614	0.1648	0.1719

relevant, with a one line reason per case. F runs supply the case facts as the query, I runs supply the LLM-generated issues, and FI runs supply both together.

5.3.1 Experiment 1: Query Formulation, Document Representation, and LLM Re-ranking

Indian legal case retrieval is a hard problem for any lexical method. Baseline MAP across all five retrieval methods falls in the range 0.15 to 0.22, a reflection of how different the vocabulary of a query case can be from the vocabulary of a 20,000-word judgment that cites long chains of statutes and precedents. Experiment 1 asks three questions. Does retrieval work better with raw case facts, with LLM-generated issues, or with both combined? Does the answer change when documents are represented as structured JSON rather than plain text? And does LLM re-ranking improve on retrieval alone, given how low the baseline MAP already is?

Part A: plain-text corpus, short issue queries.

The first part tests whether short LLM-generated issue queries improve on facts-only retrieval, across all four baselines, using the 500 plain-text judgments. Five retrieval variants are compared: facts only, each of three generated issue phrases on its own, and a Reciprocal Rank Fusion (RRF, $k=60$) of facts with all three issues.

Table 11 reports MAP for all five variants across the four baselines tested in this part. Facts-only retrieval beats every individual issue query across all four methods shown, with no exceptions. TF-IDF cosine reaches the highest facts-only MAP at 0.2049, and pure BM25 the lowest at 0.1853. No single issue phrase wins consistently across methods. Fusing facts with all three issues using RRF closes part of the gap but still falls short of facts-only retrieval on its own.

Part B: plain-text corpus, long issue queries with LLM re-ranking.

Part A suggests a cause for why issue queries under-perform: BM25 and TF-IDF match on word overlap, and an 8 to 15 word issue phrase is too sparse to match well against a 20,000-word judgment. Part B tests this directly by expanding each issue to 20 to 30 words and joining all three generated issues with OR, so that the issue query as a whole approaches the term density of the document it is being matched against. An LLM re-ranking stage is added on top of retrieval.

The pipeline works in two stages. Each retrieval method first builds a pool of 100 candidates using lexical or hybrid scoring. The LLM then reads this pool, with every document truncated to a 600-character extract so that all 100 candidates fit within a

Table 12 Best retrieval run versus best LLM re-ranking run per method, plain-text corpus.

Method	Best Ret. Run	Ret. MAP	Best LLM Run	LLM MAP	Δ MAP
BM25 (B1)	B1_FI	0.1866	B1_ILLF	0.1793	-0.0073
TF-IDF (B2)	B2_FI	0.2195	B2_F_LF	0.2159	-0.0035
TF-IDF+BM25 (B3)	B3_FI	0.2077	B3_F_LF	0.1934	-0.0143
IDF+BM25 (B4)	B4_FI	0.2151	B4_FI_LF	0.2213	+0.0062
BM25+FAISS (B5)	B5_FI	0.1864	B5_ILLF	0.1960	+0.0096

122,000-character context budget in one API call, and selects the 20 it judges most relevant. Because the LLM only ever sees these 100 candidates, any ground-truth case missing from the pool is unrecoverable regardless of re-ranking quality. Three query types, facts only, long issues, and facts plus issues, are tested across all five retrieval methods (B1 through B5), giving 15 retrieval runs and 45 LLM re-ranking runs (B1 through B5, each crossed with F, I, and FI).

Even with this longer, denser issue query, issues-only retrieval still does not close the gap with facts. What does close it is combining facts with issues: the facts-plus-issues query outperforms facts alone across all five retrieval methods, showing that issues do carry useful retrieval signal, but only once they are large enough to approach the document’s own term density. Table 12 compares the best retrieval run against the best LLM re-ranking run for each method on this corpus. LLM re-ranking does not improve MAP for three of the five methods. Only IDF+BM25 (B4) and BM25+FAISS (B5) benefit, by +0.0062 and +0.0096 respectively. The best full pipeline is facts-plus-issues retrieval re-ranked with a facts-only prompt, at MAP 0.2213. Issues-only retrieval still gives the lowest recall ceiling, roughly 14 points below facts-only.

This points to a practical problem, not just a query-design one. Real lawyers search with short, abstract questions, not 30-word padded queries. So instead of growing the query, Part C shrinks the document, using a structured summary, for three reasons. First, truncating a plain-text judgment to 600 characters cuts off content blindly, with no way to know what was lost; a structured summary instead keeps the essential fields, facts, issues, and principle, no matter where they sat in the original text. Second, a short structured summary matches how lawyers actually search, so comparing a short query against it is fairer than comparing it against 20,000 words of raw text. Third, a structured summary makes it easy to quickly tell what a case is about and whether it is actually useful, not just lexically similar.

Part C: structured JSON corpus.

Part C asks whether the problem lies in the queries or in the documents themselves. Plain-text judgments mix procedure, fact, and reasoning into one undifferentiated block, which makes it hard for term-frequency methods to find the legally relevant portion. Part C repeats the same 60-configuration ablation on 496 structured JSON documents, where each judgment is split into separate fields for facts, issues, statutes, precedents, legal principle, and key subjects.

Table 13 Best retrieval run versus best LLM re-ranking run per method, structured JSON corpus.

Method	Best Ret. Run	Ret. MAP	Best LLM Run	LLM MAP	Δ MAP
BM25 (B1)	B1_I	0.2010	B1_FILF	0.2267	+0.026
TF-IDF (B2)	B2_FI	0.2072	B2_ILF	0.2420	+0.035
TF-IDF+BM25 (B3)	B3_I	0.2021	B3_ILF	0.2302	+0.028
IDF+BM25 (B4)	B4_I	0.1838	B4_FILF	0.2379	+0.054
BM25+FAISS (B5)	B5_I	0.2183	B5_ILF	0.2644	+0.046

The result reverses the plain-text finding. Issues-only queries now outperform facts-only queries at retrieval, for all five methods, and LLM re-ranking improves MAP across all five methods as well, with the largest gains on IDF+BM25 (+0.054) and BM25+FAISS (+0.046). Table 13 shows the best retrieval run against the best LLM re-ranking run for each method on the structured corpus, the same comparison Table 12 reported for plain text. The best overall pipeline is issues-only retrieval re-ranked with a facts-only prompt, BM25+FAISS, reaching MAP 0.2644 and MRR 0.4164.

For JSON documents, the 600-character extract is built by prioritising labelled fields, principle, facts, issues, and final judgment, drawn from the structured summary. This guarantees the model sees legally decisive content rather than incidental opening text, even within the same character budget. The re-ranking prompt is otherwise unchanged from the plain-text setting. The issues query supplies lexical precision at retrieval, while the facts query gives the LLM full narrative context for re-ranking.

Despite these gains, recall ceiling remains a hard limit. Table 14 lists the five retrieval runs with the highest mean recall ceiling on the structured corpus. Even the best of these reaches only 0.7009, meaning close to 30% of ground-truth cases never enter the pool and are out of reach for any re-ranker.

Table 14 Top-5 retrieval runs by mean recall ceiling, structured JSON corpus.

Run	Method	Query Type	Mean Recall Ceiling
B5_I	BM25+FAISS hybrid	Issues only	0.7009
B3_FI	TF-IDF+BM25 fusion	Facts + issues	0.6949
B3_I	TF-IDF+BM25 fusion	Issues only	0.6699
B5_FI	BM25+FAISS hybrid	Facts + issues	0.6659
B1_I	BM25 only	Issues only	0.6566

Summary.

Two findings carry forward from Experiment 1. First, on query formulation, Table 11 and the Part B results show that issues alone are never enough on plain text, but issues do carry real retrieval signal once they are dense enough to match the document: the facts-plus-issues query beats facts alone across all five methods in Table 12, which is why a 20,000-word plain-text judgment needs a 20-to-30-word issue query just to compete, something a real lawyer’s short query would never supply on its own.

Second, on document representation, Table 12 and Table 13 tell two different stories about LLM re-ranking. On the plain-text corpus, retrieval alone already reaches a reasonably good MAP, and re-ranking on top of it barely helps, improving only two of the five methods and by a small margin. On the structured JSON corpus, retrieval-only MAP for the best method is actually lower than its plain-text counterpart, yet re-ranking lifts MAP for all five methods, by as much as +0.054. In other words, the LLM does not do its best work on the corpus where retrieval was already strong; it does its best work on the corpus that gives it a clear, labelled view of facts, issues, and principle to reason over. This shows that LLM re-ranking can meaningfully improve case law retrieval, but only once the document representation gives it something structured to work with. The remaining bottleneck is the recall ceiling of about 0.70, shown in Table 14 and partly an effect of the 600-character per-document budget, which is the problem Experiment 2 takes on.

5.3.2 Experiment 2: Union-Pool Retrieval with BM25+FAISS

Experiment 1 showed that nearly 30% of relevant judgments never reach the retrieval pool at all, which limits what any re-ranker can do. Experiment 2 tests whether combining best performed all three B5 query variants, facts only, issues only, and facts plus issues, into one deduplicated union pool can recover the cases that each query type misses on its own. Each variant retrieves its own top 100 documents, and these are merged by round-robin deduplication into a pool of up to 160 unique documents. The LLM selects the top 30 results, using one call for issue generation and three for re-ranking.

Re-running the issues-only configuration from Experiment 1 under identical settings gives a recall ceiling of 0.6723, compared to 0.7009 in Experiment 1, even with the same pool size and temperature. This gap comes from the LLM-generated issue queries themselves: even at temperature 0, small wording differences change which documents get retrieved. Run-to-run variance is therefore an expected property of any pipeline that depends on LLM-generated queries.

Table 15 compares the three individual query variants against the union pool. The union pool raises the recall ceiling to 0.7824, above both the 0.6723 re-run baseline and the original 0.7009 reported in Experiment 1, confirming that the three query types do retrieve partially different sets of relevant documents. At the same time, Table 15 shows the union pool’s retrieval-only MAP at 0.1670, lower than the issues-only MAP of 0.1816 in this same table. It is not possible to say from this comparison alone how much of that drop comes from the union pool diluting rank quality by spreading candidates across three query types, and how much simply reflects the same run-to-run variance already seen in the recall ceiling figures above. What is clear is that the union pool surfaces ground-truth cases the individual pools miss, so despite this open question on MAP, the rest of Phase 2 carries the union pool forward rather than any single query variant.

Table 16 reports LLM re-ranking results on the union pool under all three prompts. With the facts-plus-issues prompt, the union pool reaches MAP 0.2536 and MRR 0.4121, gains of +0.072 MAP and +0.137 MRR over the best retrieval-only baseline in Table 15 (B5.I, MAP 0.1816, MRR 0.2748). LLM re-ranking more than recovers

Table 15 Retrieval ceiling: individual B5 query variants versus the union pool.

Run	Pool	GT/q	RC	MAP	P@20	R@20	NDCG@20	MRR
B5_F	100	2.5	0.6097	0.1544	0.065	0.3145	0.2224	0.2743
B5_I	100	2.5	0.6723	0.1816	0.066	0.3059	0.2351	0.2748
B5_FI	100	2.7	0.6565	0.1760	0.071	0.3499	0.2483	0.2972
B5U	160	3.0	0.7824	0.1670	0.068	0.3282	0.2308	0.2793

RC = recall ceiling; GT/q = mean ground-truth docs per query.

Table 16 LLM re-ranking results on the union pool, by prompt.

Pool	Prompt	GT ctx	GT drop	MAP	P@20	R@20	NDCG@20	MRR
B5U	facts+issues	2.6	0.4	0.2536	0.072	0.3899	0.3113	0.4121
B5U	facts only	2.6	0.4	0.2456	0.078	0.3891	0.3042	0.3790
B5U	issues only	2.6	0.4	0.2244	0.067	0.3586	0.2744	0.3469

GT ctx = mean ground-truth docs in context; GT drop = mean GT docs lost to budget cap.

the precision lost at the retrieval stage. The MRR gain of +0.137 is the largest single gain across all of Phase 2, showing that the LLM ranks relevant cases substantially higher once the union pool gives it more ground-truth cases to work with.

Summary.

The union pool raises the recall ceiling well above the issues-only baseline, in Table 15, and after re-ranking, in Table 16, raises MAP and MRR past anything a single query type achieves on its own. Whether the small drop in retrieval-only MAP comes from rank dilution in the larger pool or from ordinary run-to-run variance in the LLM-generated queries remains as a question.

5.3.3 Experiment 3: Multi-Prompt RRF Ensemble

Experiments 1 and 2 used a single re-ranking prompt, one that treats shared legal issues, applicable principles, and factual similarity as equally valid signals of relevance. That prompt, on the issues-only pool, reached MAP 0.2644 in Experiment 1 (B5_ILLF) – still the best result so far. Experiment 2’s union pool, re-ranked with the same kind of prompt, only reached 0.2536. Experiment 3 asks why: is the union pool worse because spreading candidates across three query types dilutes the ranking, or because the prompt used in Experiment 2 was not well suited to the union pool? To find out, we bring the issues-only pool back into the comparison and test new prompts on both pools side by side, rather than treating Experiment 2’s union pool result as final.

Three new prompts are tested on both the issues-only pool and the union pool, then combined with RRF. Prompt 1 checks doctrinal alignment. Prompt 2 checks three things together: the same legal question, the same applicable principle, and whether a judge would treat the case as precedent. Prompt 3 matches on legal holdings rather than surface facts. All other settings match Experiment 1: top 100 pool, LLM selects the top 20, greedy decoding, 600-character document cap per candidate.

Table 17 Retrieval baselines: B5 variants and the union pool.

Run	Pool	RC	GT/q	MAP	P@20	R@20	NDCG@20	MRR
B5_F	100	0.5828	2.4	0.1244	0.052	0.2733	0.1782	0.2053
B5_FI	100	0.5990	2.4	0.1536	0.062	0.3177	0.2159	0.2421
B5_I	100	0.6324	2.5	0.2070	0.069	0.3757	0.2783	0.3296
B5U	160	0.7439	3.0	0.1604	0.070	0.3414	0.2279	0.2413

RC = recall ceiling; GT/q = mean ground-truth docs per query.

Table 18 All LLM runs: individual prompts and RRF ensembles on the issues-only and union pools.

Run	Pool	Avg Hits	MAP	P@20	R@20	NDCG@20	MRR
B5U_RRF	B5U	1.4	0.2216	0.069	0.3640	0.2997	0.3849
B5_I.P1	B5_I	1.4	0.2193	0.068	0.3488	0.2949	0.3865
B5U.P3	B5U	1.2	0.2104	0.060	0.3359	0.2836	0.3737
B5U.P1	B5U	1.2	0.2074	0.060	0.3155	0.2750	0.3468
B5_I.P3	B5_I	1.3	0.2072	0.065	0.3376	0.2776	0.3456
B5_I.RRF	B5_I	1.5	0.2004	0.074	0.3807	0.2846	0.3343
B5_I.P2	B5_I	1.3	0.1979	0.064	0.3472	0.2762	0.3483
B5U.P2	B5U	1.3	0.1946	0.063	0.3397	0.2696	0.3268

For LLM re-ranking, only the issues-only pool and the union pool are carried forward, since Table 17 shows these are the two strongest retrieval baselines.

Summary.

None of the new prompts beat Experiment 1’s original result. The best result here is B5U_RRF at MAP 0.2216, still well below 0.2644. The closest result on a single pool is B5_I.P1 at 0.2193, close to the union-pool’s best but using the smaller, less diluted issues-only pool. This suggests the original prompt’s broader approach – treating factual similarity as a valid signal alongside legal principle – worked better than these three stricter, principle-only prompts.

RRF only helps on the larger pool. On the union pool, RRF (0.2216) beats every individual prompt tested on that pool (best single: 0.2104). On the issues-only pool, RRF lowers MAP (0.2004 vs. 0.2193 for the best single prompt, P1) but raises recall (R@20: 0.3807 vs. 0.3488). The reason is that P1 already dominates most queries on this pool, so blending it with weaker prompts costs precision even as it recovers a few extra relevant cases.

No single prompt wins on both pools. P1 is best on the issues-only pool (0.2193); P3 is best on the union pool (0.2104). The best prompt depends on which pool it runs on. This also shows up in how often each prompt wins per query: on the issues-only pool, P1 wins 24% of the 50 queries, P3 wins 18%, P2 wins 12%. On the union pool the spread is more even: P3 wins 22%, P2 wins 20%, P1 wins 18%. Overall, RRF matches or beats the best single prompt on 54% of issues-only queries and 56% of union-pool queries.

Table 19 Retrieval pools versus cross-encoder filtered pools.

Run	Type	Pool	RC	GT/q	MAP	P@20	R@20	NDCG@20	MRR
B5.I	retrieval	100	0.658	2.3	0.1729	0.067	0.3797	0.2506	0.2874
B5U	retrieval	162	0.789	3.0	0.1448	0.066	0.3282	0.2101	0.2293
CE_B5.I	CE-filtered	40	0.402	1.5	0.1147	0.057	0.2747	0.1827	0.2081
CE_B5U	CE-filtered	40	0.330	1.4	0.1098	0.053	0.2613	0.1726	0.1945

RC = recall ceiling; GT/q = mean ground-truth docs per query.

Table 20 Cross-encoder impact on MAP and R@20, all configurations.

Configuration	MAP		Δ	R@20		Δ
	No-CE	CE		No-CE	CE	
P1 on B5.I	0.2310	0.1553	-0.076	0.4125	0.2976	-0.115
P2 on B5.I	0.1916	0.1136	-0.078	0.3596	0.2568	-0.103
P3 on B5.I	0.2335	0.1591	-0.074	0.3825	0.3189	-0.064
RRF on B5.I	0.2376	0.1392	-0.098	0.4184	0.3093	-0.109
P1 on B5U	0.2055	0.1525	-0.053	0.3318	0.2771	-0.055
P2 on B5U	0.1763	0.1266	-0.050	0.3163	0.2375	-0.079
P3 on B5U	0.2285	0.1241	-0.104	0.3643	0.2631	-0.101
RRF on B5U	0.2168	0.1365	-0.080	0.3731	0.2837	-0.089

A caution on comparing these numbers. Issue queries are LLM-generated, so the retrieved pool itself can vary slightly between runs even at temperature 0. Small MAP differences across experiments, including the comparisons above, should be read as indicative rather than as exact, fixed rankings.

5.3.4 Experiment 4: Cross-Encoder Pre-filtering

The earlier experiments left two problems unsolved. The LLM sometimes misses relevant cases even when they are already in the pool. And a bigger pool only helps if the LLM is allowed to pick more cases from it. Experiment 4 tests a possible fix: filter the pool down to a smaller, cleaner set of documents before the LLM sees it.

We used a cross-encoder for this filtering step. We first tried BAAI/bge-reranker-base on 10 queries, but its rankings were weak, so we switched to cross-encoder/ms-marco-MiniLM-L-6-v2 (22M parameters) for the full test. This model scores each document against the facts query and keeps only the top 40. The filtered pool is then re-ranked the same way as in Experiment 3.

Summary.

The filter made things worse every time, with no exceptions. Recall ceiling dropped from 0.658 to 0.402 on the issues-only pool, and from 0.789 to 0.330 on the union pool. That means most ground-truth cases were thrown out before the LLM ever saw them. MAP dropped by 0.05 to 0.104, and R@20 dropped by 0.055 to 0.115, in every configuration. The likely reason: this cross-encoders was not trained on Indian legal

Table 21 The four structured JSON formats compared in Experiment 5.

Format	Fields kept	Fields	Docs fitting
V1	All 21 fields	21	98
V2	Facts, issues, reasoning, new legal principle, application, important subjects discussed	6	255
V3	Facts, issues, citation names (statutes and precedents), new legal principle, important subjects discussed	6	292
V4	Facts, issues, citation principles (statutes and precedents), new legal principle, important subjects discussed	6	224

V3 lists citation names with no explanation of why they matter. V4 replaces names with the distilled legal rule behind each citation.

text, so it doesn’t know how to judge relevance for Indian Supreme Court cases. Cross-encoder filtering should not be used here unless it is first fine-tuned on Indian legal text.

5.3.5 Experiment 5: Different Document Fields

This experiment is motivated by a finding from Phase 1 (Section 5.2). There, including the full Precedents Referred field (with both its subfields) dropped MAP from 0.365 to 0.313 and MAR from 0.672 to 0.524, because the long precedent and statute names added noise and ate up context space. This raises an open question for Phase 2’s structured-summary corpus: does a citation field help once it states the legal principle behind each citation instead of just the name, and does the choice of fields more broadly affect how many ground-truth cases survive into the pool in the first place. Experiment 5 tests both questions directly.

Experiment 5 tests whether a better choice of document fields lets more ground-truth cases survive into the pool, independent of pooling strategy. Retrieval depth is raised to 150 documents and LLM re-ranking selects the top 30. Four structured JSON formats are compared (Table 21): V1 keeps all 21 fields, fitting only 98 documents in context; V2, V3, and V4 each keep six fields and fit the full pool with zero truncation, differing only in what they say about citations and reasoning. V2 keeps full reasoning, V3 lists citation names only, and V4 replaces citation names with the distilled legal principle behind each citation.

For each format, retrieval is run under a short and a long issue query on both the issues-only pool (B5_I.short, B5_I.long) and the union pool (B5U.short, B5U.long), each retrieving the top 150 documents. LLM re-ranking is run only on the union pool, not on the issues-only pool, since the goal is to test ranking quality once coverage is maximised.

Summary.

Issue query length. Neither length wins outright. At retrieval, the long query gives the best MAP in all four formats (Table 22, B5_I rows) on the issues-only pool, while the short query gives the best recall ceiling on the union pool in all four formats (B5U

Table 22 Retrieval performance, short versus long issue queries, both pools, across all four document formats.

Fmt	Run	Ceil.	MAP	P@20	R@20	NDCG@20	MRR
V1	B5.I.short	0.7180	0.1833	0.065	0.3134	0.2368	0.2769
V1	B5.I.long	0.7544	0.2178	0.076	0.3770	0.2835	0.3124
V1	B5U.short	0.8495	0.1676	0.067	0.3229	0.2275	0.2722
V1	B5U.long	0.8235	0.1830	0.077	0.3686	0.2560	0.2923
V2	B5.I.short	0.7238	0.1799	0.069	0.3458	0.2453	0.2935
V2	B5.I.long	0.7450	0.1928	0.075	0.3842	0.2660	0.2912
V2	B5U.short	0.8394	0.1517	0.064	0.3062	0.2061	0.2304
V2	B5U.long	0.8235	0.1528	0.062	0.2806	0.2019	0.2378
V3	B5.I.short	0.7361	0.1843	0.070	0.3478	0.2491	0.2956
V3	B5.I.long	0.7454	0.1988	0.076	0.3875	0.2713	0.3062
V3	B5U.short	0.8592	0.1544	0.068	0.3142	0.2114	0.2314
V3	B5U.long	0.8188	0.1562	0.065	0.3008	0.2097	0.2393
V4	B5.I.short	0.7112	0.1907	0.063	0.3343	0.2518	0.3121
V4	B5.I.long	0.7108	0.1935	0.071	0.3739	0.2673	0.3133
V4	B5U.short	0.8352	0.1582	0.063	0.3196	0.2178	0.2495
V4	B5U.long	0.7946	0.1549	0.065	0.3112	0.2140	0.2460

Bold = higher MAP (B5.I rows) or higher ceiling (B5U rows) within each format.

rows). After re-ranking (Table 23) on the union pool, the winner flips by format: short wins for V1 and V4, long wins for V2 and V3. The best length depends on the document format, not on a universal rule. For practical deployment, V4 with a short query is the best choice: it gives the best overall MAP, costs almost nothing in precision compared to a long query, and matches how lawyers actually search.

Document format. V4 is best. Ranking the four formats by their best re-ranked MAP gives V4 (0.3124) > V2 (0.2988) > V1 (0.2894) > V3 (0.2885). A citation’s name alone (V3) adds no benefit over dropping citations entirely (V2); stating the legal principle behind each citation (V4) gives the best MAP in the whole study. Phase 1’s finding was that the Precedents Referred field lowers MAP. Here, we test more precisely which part of that field is responsible. Phase 1 used all three sub-fields of Precedents Referred (name, principle, and application); Phase 2 uses only one sub-field, principle, and this single field helps improve MAP.

Pool size versus field content. Fitting more documents into the union pool does not improve the result. V3 fits the most documents (292) of any compact format but gives the worst MAP (0.2885); V4 fits fewer documents (224) than both V2 (255) and V3 (292) but gives the best MAP (0.3124). Pool size is not what drives the gain. What matters is what each field says: V4’s gain over V1, V2, and V3 holds specifically because its citation field states what each citation means rather than only naming it (V3) or omitting it (V2).

5.3.6 Experiment 6: Union Pool Construction and Rank Quality

Experiment 2’s failure analysis raised a question worth testing directly. Facts-plus-issues retrieval adds nothing new: it finds the same cases the separate facts and issues

Table 23 LLM re-ranking on the union pool, both union pool variants and both prompts, across all four document formats.

Fmt	Pool	Query	MAP	P@20	R@20	NDCG@20	MRR
V1	B5U_short	LF	0.2894	0.131	0.7006	0.4214	0.3734
V1	B5U_long	LF	0.2479	0.120	0.5998	0.3593	0.3438
V1	B5U_short	LI	0.2738	0.128	0.6875	0.3966	0.3538
V1	B5U_long	LI	0.2646	0.109	0.5013	0.3434	0.3630
V2	B5U_short	LF	0.2776	0.120	0.5774	0.3759	0.3530
V2	B5U_long	LF	0.2988	0.135	0.6847	0.4179	0.3505
V2	B5U_short	LI	0.2913	0.138	0.7098	0.4140	0.2922
V2	B5U_long	LI	0.2792	0.118	0.6027	0.3766	0.3445
V3	B5U_short	LF	0.2763	0.115	0.6492	0.3925	0.3529
V3	B5U_long	LF	0.2885	0.123	0.6683	0.4053	0.3538
V3	B5U_short	LI	0.2856	0.131	0.6779	0.4065	0.3285
V3	B5U_long	LI	0.2882	0.139	0.6350	0.4008	0.3684
V4	B5U_short	LF	0.3077	0.119	0.6788	0.4220	0.3738
V4	B5U_long	LF	0.2891	0.133	0.6519	0.4039	0.3633
V4	B5U_short	LI	0.3124	0.139	0.7392	0.4427	0.3379
V4	B5U_long	LI	0.2903	0.147	0.7064	0.4198	0.3197

Bold = best MAP within each format. V4 with the short issue query and issues-only prompt gives the best MAP in all of Phase 2. LF = facts-only prompt; LI = issues-only prompt.

Table 24 Retrieval ceiling and recall, issues-only pool versus union pool, short and long issue queries.

Run	Ceiling	Avg GT Hits	MAP	R@20
B5.I.short	0.7060	2.7	0.1834	0.3895
B5.I.long	0.7136	2.9	0.2050	0.3584
B5.U.short	0.8430	3.4	0.1634	0.3549
B5.U.long	0.8109	3.3	0.1803	0.3375

queries already find between them. Worse, fusing queries with RRF can push a correct case down to a low rank even when each query individually ranked it well, because RRF re-ranks by combined evidence rather than keeping either query’s own order.

Experiment 5 already hinted at this. There, the issues-only pool got a better MAP than the union pool, even though the union pool had a higher recall ceiling and found more ground-truth cases. In other words: the union pool finds more, but ranks worse; the issues-only pool finds less, but ranks better. Experiment 6 checks whether this trade-off holds after LLM re-ranking too, by running both pools through the same pipeline and comparing them directly. As a second check, the exact union-pool setup from Experiment 5 is re-run under identical settings with V4 document format, to see how much the results vary from run to run before trusting any comparison between pools.

Table 25 LLM re-ranking, issues-only pool versus union pool, short and long issue queries.

Run	MAP	NDCG@10	MRR	Avg Hits
B5_I_short_LI	0.2916	0.3988	0.2978	0.7
B5_I_short_LF	0.3020	0.4011	0.3892	0.8
B5_I_long_LI	0.3091	0.4138	0.3483	0.7
B5_I_long_LF	0.3080	0.4156	0.3829	0.8
B5U_short_LI	0.2770	0.3294	0.3168	0.9
B5U_short_LF	0.2724	0.3127	0.3112	0.9
B5U_long_LI	0.2669	0.3174	0.3255	0.7
B5U_long_LF	0.2661	0.3046	0.3147	1.0

Summary.

Run-to-run variance is real, but the ranking is stable. Re-running the Experiment 5 union-pool setup under identical settings gives MAP 0.2770, about 0.035 below the 0.3124 reported earlier for that same setup. Despite this drop, the order of the four configurations is unchanged: short query with the issues-only prompt still beats short with facts, which still beats long with issues, which still beats long with facts – the same order as in Experiment 5, just at lower values. This is treated as ordinary noise in an LLM-based pipeline, not a changed finding: the values shift, but which setting wins does not. Any MAP comparison between pools in this study should still be read with a margin of roughly ± 0.03 to 0.04 in mind.

The issues-only pool ranks better than the union pool, at both stages. At retrieval, the issues-only pool reaches a higher MAP than the union pool (0.2050 vs. 0.1803), even though the union pool has a higher recall ceiling (0.843 vs. 0.714) and finds more ground-truth cases on average (3.4 vs. 2.9): merging two query types into one pool gains coverage but costs more in rank quality than that coverage is worth. This holds, and grows, after LLM re-ranking. The issues-only pool beats the union pool in all four matched settings (short/long query, facts/issues prompt), with the best result overall at MAP 0.3091 (long query, issues-only prompt) and it also hands the LLM a smaller, cleaner shortlist (5–9 documents) than the union pool does (14–16), consistent with the LLM doing better from a well-ranked list than from a larger pool whose order RRF has already disturbed.

5.3.7 Experiment 7: Compressed Facts as Query

Experiments 1 and 5 point to opposite query strategies depending on the document type: long facts queries work on plain text, short issue queries work on the structured V4 corpus. Neither experiment tested a compressed facts query, where the raw case facts are shortened by an LLM before retrieval. This matters because the project’s broader goal is to move toward issue-based retrieval, and if a short, compressed facts query could match or beat the issues query, that would suggest facts compression is a workable alternative, not just issues. A Stage 0 LLM call reduces the raw query facts to a 4–5 sentence summary, roughly half of the original length. This compressed text replaces the raw facts wherever facts are used: the facts-only retrieval leg, the facts-plus-issues leg, and the re-ranking prompt. Experiment 7 compares three queries

Table 26 Retrieval baselines and LLM re-ranking with compressed facts, V4 structured format (Experiment 7, 50 AILA queries).

Run	MAP	R@20	NDCG@20	MRR	Avg Hits
<i>V4 structured corpus, B5U pool</i>					
B5_F	0.1915	0.3464	0.2538	0.3013	2.96
B5_I	0.1857	0.3716	0.2530	0.2758	2.72
B5_FI	0.1979	0.3826	0.2695	0.3008	3.04
B5U_short	0.2013	0.3590	0.2647	0.3067	3.44
LLM_LF	0.2892	0.6676	0.4072	0.3843	0.90
LLM_LI	0.2834	0.6121	0.3867	0.3505	0.68
LLM_LFI	0.3143	0.6500	0.4196	0.4004	0.84

LF = compressed-facts prompt; LI = issues prompt; LFI = compressed facts + issues prompt.

against the V4 union pool across all 50 AILA queries: compressed facts (F), the short issues query carried over from Experiment 5 (I), and their combination (FI).

Summary.

Compressed facts beat issues at retrieval (B5_F MAP 0.1915 versus B5_I’s 0.1857), reversing Experiment 5’s result with raw facts. At re-ranking, LF and LI are close on average (0.2892 versus 0.2834), but this hides a failure pattern: on a recurring subset of queries (Q20, Q30, Q35, Q38, Q40), LF over-selects (26–30 of 30 candidates) while LI selects a narrow 1–12 and wins decisively (Q40: LI AP 0.267 versus LF 0.054). Compressed facts retain generic procedural detail with no sharp hook to filter on, while the issue query names the specific legal question. Compressed facts help at retrieval but are fragile at re-ranking; pairing them with issues is what holds up at both stages.

Two costs of compressing the facts query.

First, an extra LLM call: generating issues from raw facts already costs one call, and compressing the facts query costs a second one. Second, the structured summary represents facts in 4–5 sentences rather than the raw judgment’s 10–14, so matching that density means compressing the query to a similarly short form. At that length, wording is loosely constrained, and repeated compression calls on the same facts can produce different phrasing each run. Since this text feeds retrieval and re-ranking directly, the resulting wording variation can change which documents are retrieved and selected from run to run, independent of the underlying case — a source of instability that raw facts and short issues do not carry to the same degree. A second run of Experiment 7 under identical settings bears this out: the facts-versus-issues ordering reverses outright at retrieval (B5_F MAP 0.1915 $\rightarrow$ 0.196 versus B5_I MAP 0.1857 $\rightarrow$ 0.2152, a swing more than three times the original gap), and moves in the same direction at re-ranking (LLM_F_LF 0.2892 $\rightarrow$ 0.2413 versus LLM_I_LF 0.2834 $\rightarrow$ 0.2468). This is the scale of instability the compressed-facts query’s loosely constrained wording would be expected to produce, well beyond the ± 0.03 –0.04 MAP noise band already established for fixed-phrase queries in Experiment 6.

5.3.8 Experiment 8: Run-to-Run Consistency Check

Every configuration in this study relies on an LLM at some stage, but none of the preceding experiments directly tested how stable the pipeline’s results are when the exact same run is repeated. Experiment 8 checks this, and asks a second question alongside it: does short or long LLM-generated text give more stable results? The short and long issue queries are well suited to this, since both are already known to perform well (Experiment 5) but sit at opposite ends of the length spectrum.

The full Experiment 5 pipeline (document format V4) was run twice under identical settings on a 10-query subset of the benchmark. The two runs were compared at retrieval and at LLM re-ranking, using the drift measure defined in Section 5.1 (Equation 18).

Retrieval is highly stable. Across all retrieval configurations, 95.5% of results matched exactly between the two runs.

Short issue text is more stable than long, at retrieval. With short issues, every result matched exactly (100%). With long issues, only 90% matched, a statistically significant gap ($p = 1.5 \times 10^{-7}$). At re-ranking, this gap narrows and is no longer significant (64% vs. 61% match, $p = 0.17$).

LLM re-ranking is the main source of instability. Only 62.5% of re-ranking results matched between runs, far lower than retrieval’s 95.5%. The immediate cause is how many documents the LLM selects from the pool, which differs by 8–10 documents (out of up to 30) between runs, for both short and long issues alike – so query length on its own does not explain the size of this drift. Part of this instability likely also traces further upstream: the issue queries themselves are LLM-generated, so a fresh run can produce slightly different issue text, which changes which documents enter the pool, which in turn gives the re-ranking LLM a different set of candidates to choose from each time.

Summary.

Retrieval is reliable and repeatable; LLM re-ranking is not, mainly because the LLM does not consistently select the same number of documents each time. This means the headline best result in Phase 2 (MAP 0.3124, Experiment 5) is a point estimate, not an exact number: a re-run could reasonably land ± 0.03 to 0.04 away from it, the same margin already seen in Experiment 6.

5.3.9 Failure Analysis

We look at the best issue-retrieval configuration (Exp. 5, format V4) to see where the pipeline holds up and where it doesn’t. Three failure patterns stand out.

Facts vs. Issues: Complementary Signals

Comparing the ground-truth cases retrieved by B5_F (facts), B5_I (issues), and B5_FI (combined) shows the two signals aren’t interchangeable. Facts retrieval finds a ground-truth case that issues retrieval misses completely in 9 queries (Q41, Q34, Q43 among them), and issues retrieval finds one that facts misses in 16 queries (Q1, Q32, Q29 among them). Facts-based retrieval picks up on shared textual specifics, things

like party type, statute, or fact pattern, while issues-based retrieval picks up on a shared legal question even when the facts on the ground look nothing alike. That second property is really what legal retrieval needs, since a precedent can matter simply because it settled the same point of law, regardless of how different the underlying events were. The two signals are covering largely different ground, so using both together is what gets you full coverage.

Retrieval Failures: Zero-Hit Queries

Only two queries come back with zero ground-truth hits under every retrieval signal and every retrieval-reranking combination we tested, Q3 and Q24. Since nothing recovers even a partial hit for these, the problem sits upstream of both retrieval and reranking. Looking closer, the two queries fail for different reasons.

Q3 is about food adulteration, but the case it's labelled against, C1, is a citizenship and issue-estoppel matter. There's no real connection between them, not even an indirect one. No retriever could have found this, because there's nothing there to find. This looks like a labelling error in the dataset rather than a system failure, and should be set aside when judging retrieval quality.

Q24 fails for a completely different reason. The issue-generation step produced something unusable here: three of the four issue fields are empty, and the one that isn't is just a restatement of the facts rather than an actual legal issue. With nothing meaningful to search on, every retrieval signal comes up empty.

Selection Failures: Retrieval Succeeds, the LLM Does Not

Q46 is the clearest case where retrieval succeeded but the batched reranking run still failed completely. Eight of its nine ground-truth cases reach the pool through the facts signal alone, and the issues signal surfaces all nine, giving the reranker a clean shot at every relevant case — yet in the original batched run, the LLM selected zero of nine.

Rerunning Q46 individually told a different story. With the same retrieval ceilings (0.889–1.000) and the same pool, the LLM now selected 4, 3, and 4 of 9 ground-truth cases across the three query views (NDCG@20 of 0.373, 0.488, and 0.397). The right cases were reachable all along; the batch run simply failed to find them, pointing to the same kind of parallel-processing fragility behind the C71/C342/C406/C481 summary-generation failures, not a deeper flaw in the reranking step itself.

That said, the individual run was still imperfect, and its own selection pattern shows why. One justification recurs constantly: a High Court shouldn't overturn an acquittal just because another view is possible. This principle is broad enough to fit almost any criminal appeal, and it crowds out several ground-truth cases that should have ranked higher.

5.3.10 Discussion

The eight Phase 2 experiments build on one another and together support one central idea, from which the remaining findings follow.

Relation between query density and document density.

Query density must match document density at the similarity-retrieval stage. Lexical and embedding retrieval score by overlap, so a query far sparser than the document cannot match it: an 8–15 word issue query fails against a 20,000-word judgment, while expanding it (Exp. 1, Part B) or using the full facts query recovers performance, and the same short query wins once the document itself is reduced to a structured summary (Exp. 1, Part C).

At the LLM stage the constraint is different: information sufficiency, not symmetric density. Re-ranking is a reasoning step, so query detail and document detail substitute for each other. Where the document already states its legal point, as in V4’s principle field, a short query suffices and issue queries win; where the format leaves a gap, whether from unfiltered raw text (V1), unexplained reasoning (V2), or bare citation names (V3), a longer query and the facts-only prompt compensate (Exp. 5). This is why no single query length wins at the LLM stage, and why document representation, not query length, is the stronger lever.

Experiment 7 points the same way from the facts side: a compressed facts query, brought closer to the structured summary’s density, beats the raw issue query on the V4 corpus. We do not treat this as a confirmed result, however, and state the reason explicitly rather than let the comparison stand unqualified. Unlike the short, fixed-phrase issue queries used elsewhere in this study, the compressed facts text is generated fresh by the LLM at each run, and it is considerably longer than a short issue query. Both properties raise uncertainty: Experiment 8 already shows that long, freely-worded generated text is substantially less reproducible than short generated text, so a single run on compressed facts carries more run-to-run uncertainty than a short issue query does, and a one-off comparison cannot separate a genuine density effect from this run’s particular phrasing happening to retrieve well. Confirming the effect properly would require fixing one set of facts and issue queries for each case and re-running retrieval multiple times to measure the variance directly, the same design used in Experiment 8 for issue length. We did not run this check for compressed facts, since it would have required more LLM calls than our WatsonX API budget allowed, so Experiment 7’s result is reported here as a single-run observation consistent with the density hypothesis, not as confirmation of it. The experiments overall fix the direction of the density effect at retrieval on the firmer, multi-configuration grounds of Experiment 1, not a precise ratio, since query length was tested only at discrete levels; locating that threshold, and properly testing the compressed-facts case with repeated runs, are both left to future work.

Document representation is the stronger lever.

Representation changed outcomes more than query or prompt choice did. Moving from plain text to structured JSON (Exp. 1) reversed which query type won at retrieval and turned LLM re-ranking from a step that mostly hurt MAP into one that helped all five methods. Within structured formats, field choice alone produced a best-MAP spread of 0.2479 to 0.3124 (Exp. 5). Citation fields were the clearest case: bare citation names added little — at points worse than dropping citations entirely — while stating the principle behind each citation gave the best results in Phase 2.

Re-ranking pays off only when the document supports it.

On plain text, where retrieval-only MAP was already comparatively strong, re-ranking improved only two of five methods. On structured JSON, where the best method’s retrieval-only MAP was lower, re-ranking improved all five, by up to +0.054. A strong retrieval score does not predict the re-ranking gain; document structure does.

Coverage and rank quality trade off.

Pooling that increased ground-truth coverage did not automatically improve ranking. The union pool (Exp. 2) raised the recall ceiling and produced the study’s largest MRR gain, yet its retrieval-only MAP was below the issues-only pool’s, and Exp. 6 confirmed the issues-only pool beats the union pool on MAP at both stages despite a lower ceiling and fewer hits. Fusing query legs can push well-ranked documents down even when each leg ranked them well. Coverage and rank quality are a genuine trade-off, not something pooling resolves.

Prompt and format choices do not transfer.

The best re-ranking prompt depended on the pool (Exp. 3) and on the document format (Exp. 5); no single prompt or query length won everywhere. Formats that state their legal point directly favour a short matching query; formats that leave a gap favour a longer query and the facts prompt to compensate. Prompt and format must be tuned together.

Imported components and extra steps did not transfer.

The cross-encoder result (Exp. 4) is among the study’s most robust: a general-purpose MS-MARCO cross-encoder was uniformly harmful across all eight prompt×pool combinations, cutting the recall ceiling by 38.9% to 58.2% and MAP by 0.050 to 0.104 — effects far larger than the noise band and uniform in sign — most plausibly because the model was never trained on Indian legal text. Compressed facts (Exp. 7) are a weaker and more fragile case. They gave good results on the most compact format and require an extra LLM call on top of issue generation, but the deeper objection comes from Exp. 8: short LLM-generated queries are significantly more reproducible than long ones, and compressed facts produce a comparatively long, freely-worded text whose phrasing varies from run to run. Since that text feeds both retrieval and re-ranking directly, the variation propagates into which documents are retrieved and selected, so even a favourable single-run result carries more run-to-run uncertainty than the short issue query does. We therefore do not treat the compressed-facts gain as a reliable improvement; a concise, human-written statement of facts — stable across runs and free of the extra call — is the more dependable way to obtain the same density benefit.

Run-to-run variance is structural.

Because issue generation is itself an LLM call, identical configurations differ on re-run even at temperature 0. Experiment 8 locates this variance precisely: retrieval is highly reproducible (95.5% of per-query metrics identical across runs, Pearson $r = 0.980$), and within retrieval short issue text is significantly more stable than long ($p = 1.5 \times 10^{-7}$).

LLM re-ranking is the dominant source of instability (only 62.5% of metrics identical, $r = 0.755$), driven by the number of documents the model selects rather than by query representation.

6 Result Analysis and Findings

Phase 1 (Gemini Pro, 14 queries, 50 judgments) used the FIRE 2019 rank-unaware protocol, MAP and MAR. Phase 2 (Llama 3 70B, 50 queries, 496 documents) is ranking-aware, scored with MAP and MRR. The two are read together by direction, not by treating their raw MAP values as the same scale, since neither the corpus size, the query set, nor the scoring rule matches between them. What follows is organised around how the two phases relate to each other on each point, rather than metric by metric: where they agree and reinforce one another, where Phase 2 extends or corrects something Phase 1 could only show in outline, and where a real disagreement or open question remains. Mechanism, why a result came out the way it did, is covered in the Phase 1 and Phase 2 Discussion sections and is not repeated here.

6.1 Where the two phases agree

The clearest and most load-bearing agreement concerns the Precedents Referred field. Phase 1 found that including it lowered MAP from 0.365 to 0.313 and MAR from 0.672 to 0.524. Phase 2’s Experiment 5 reaches the same conclusion under a different model, a different evaluation protocol, and a larger query set (50 queries against Phase 1’s 14): a citation-free format (V2) beats one that lists citation names alone (V3), while a format that states the legal principle behind each citation (V4) beats both. A finding that survives a change of model, protocol, and scale by this much is treated as the paper’s most robust conclusion: listing what was cited does not help an LLM reranker; explaining what the citation means does.

Experiment 7’s compressed-facts query does reach strong results on the V4 format, but we do not adopt it as the recommended configuration. A compressed facts query is generated text whose length varies run to run, and that variability is inherently larger than what issue generation produces: issue generation tends to converge on a small set of dominant legal issues from a given fact pattern, whereas compressing a fact narrative down to the document’s density admits many more equally valid phrasings and lengths. This makes the compressed-facts query a less stable retrieval signal than the issues query, even where it happens to score well. Experiment 5’s V4 configuration is therefore treated as the best version overall.

Experiment 6 supports this choice. We rebuilt Experiment 5’s setup with the same union pool alongside a newly constructed issues-only pool, and re-ran the full pipeline. Although the absolute scores differ between this run and Experiment 5, as expected from the run-to-run variance already noted, the underlying finding is unchanged: in both runs, the pool retrieved with the short issues query, re-ranked using the short issues query, wins. We attribute this to the relative stability of short issue queries across generations, a stability that a lengthy, more variable facts query cannot be expected to match.

The two phases also agree, independently, that no single query strategy is sufficient on its own. Phase 1’s question-guided retrieval outperformed retrieval on raw facts and was rated useful by legal experts, but its issue-generation step produced different legal questions on different runs, a problem Phase 1 worked around by keeping only the most relevant generated issues. Phase 2’s failure analysis gives this the same shape at much larger scale: facts and issues retrieve largely *different* ground-truth cases (9 facts-only, 16 issues-only), because issues reach precedents that share a legal question despite dissimilar underlying facts. Both phases land on the same practical conclusion from different evidence: issues are valuable but not sufficient alone, and some combination with facts is needed for full coverage.

Both phases also independently support the basic premise of the whole approach: structured summaries, not fine-tuning, are what make LLM case-law retrieval workable over realistic document lengths. Phase 1 showed this could work at all, reaching MAP 0.365 on Gemini Pro with no fine-tuning. Phase 2 reached MAP ≈ 0.31 on Llama 3 70B with the same constraint, after a roughly ten-fold increase in corpus size and a strictly harder, ranking-aware scoring rule. That the result holds in the same range under a harder test, rather than collapsing, is itself part of the finding.

6.2 Where Phase 2 extends or corrects Phase 1

Several of Phase 1’s findings turn out to be one part of a larger picture that only became visible once Phase 2 tested more than a single configuration.

Phase 1’s structured-summary template was fixed: one format, with a Precedents Referred field that could be switched on or off. This was enough to show that the field added noise, but not enough to ask why, or whether a different way of representing the same citations might work better. Phase 2’s Experiment 5 reopens exactly this question by testing four different field sets on the same underlying corpus, and finds that the answer was never simply “drop the field”: a citation field that states the legal principle behind each citation (V4) outperforms both a citation-free format (V2) and a format that lists citation names without explanation (V3). Phase 1’s binary choice, precedents in or out, turns out to have been a coarse version of a finer distinction Phase 2 resolves: it is not the presence of citations that hurts, but an uninformative representation of them.

Phase 1’s finding that issue-guided retrieval beats facts-based retrieval also turns out to be conditional rather than universal. Phase 2’s Experiment 1 tested both query types on plain-text judgments as well as on structured summaries, and found that facts beat every issue query on plain text, since a short phrase cannot match a 20,000-word document; only on structured summaries does the result reverse, with issues winning across all five retrieval methods tested. Phase 1 had only ever tested the structured-summary condition, so its result was correct as far as it went, but it could not have shown that the advantage depends on document representation rather than holding on its own.

Phase 1’s qualitative observation that an ensemble retriever, FAISS and BM25 combined, surfaced judgments that either retriever missed on its own is the conceptual ancestor of two things tested formally in Phase 2: the BM25+FAISS hybrid baseline

used throughout (B5), and the union pool of Experiment 2, which fuses three differently formulated queries rather than two retrievers. Phase 2 puts a number on what Phase 1 could only describe: the union pool raised MRR from 0.2748 to 0.4121, a gain of +0.137, the largest single MRR gain in the study, by getting one correct case to rank earlier rather than by improving the full ranked list, a distinction Phase 1’s metrics could not have shown even if it had tried.

Phase 1 also showed a risk it couldn’t fully study: one of fifteen queries was blocked by Gemini Pro’s safety filter. Phase 2 avoids that problem by switching to Llama 3 70B, but hits a similar one elsewhere — four of 500 judgments failed during summary generation because of batch-processing errors (gateway timeouts and a parsing bug), confirmed when retrying them one by one worked. That fix came too late to redo the experiments, since the API quota ran out first. One of the failed judgments is the only ground truth for two queries, capping their recall throughout.

6.3 Where the phases speak past each other

Not every Phase 1 finding was picked up or tested again in Phase 2, and at least one practical conclusion is Phase 1’s alone in this paper.

Phase 1’s reading of the Precedents Referred result went further than the benchmark numbers: while dropping precedents wins on MAP, keeping them brings a larger total number of judgments connected to the case’s facts into the pool, since the additional matches are drawn from precedents the judgment itself relies on, even where some are only indirectly relevant in terms of statutes or shared facts. Phase 2 does not revisit this practical argument; its Experiment 5 only measures benchmark MAP under different field representations, not the real-world usefulness of a wider, precedent-inclusive pool. The two phases therefore agree on the benchmark result and its likely cause, but the further claim that the broader pool is worth the benchmark cost in practice rests on Phase 1’s qualitative judgement alone, and would need its own test at Phase 2’s scale to confirm.

Phase 1 also proposed, but did not build, a version of the pipeline where a lawyer chooses one legal issue from the generated list rather than having the system commit automatically to the three most relevant issues. This was a stated direction for practical deployment in Phase 1, not a tested configuration, and Phase 2 does not implement or evaluate it either. It remains open for either phase’s pipeline.

Finally, the two phases are not a controlled comparison of model choice. Phase 1 used Gemini Pro throughout; Phase 2 used Llama 3 70B throughout. Differences between the two phases’ results may reflect the change in corpus size, query count, document representation, or evaluation protocol, any of which changed at the same time as the model, so no claim in this paper isolates model choice as the cause of any particular difference between Phase 1 and Phase 2.

6.4 Research Questions Answered

Q1: How well do LLMs perform case law retrieval under context length restrictions and without fine-tuning? Well, and the result holds at scale. With no fine-tuning, Phase 1 reached MAP 0.365 on a 50-judgment corpus, and Phase 2

reached $\text{MAP} \approx 0.31$ on a 496-document corpus under a stricter, ranking-aware metric. Structured summaries are what make this possible: they keep each judgment short enough to fit in the LLM’s context window without retraining the model on legal text.

Q2: How do structured summaries contribute to LLM-based case law retrieval? They make retrieval and re-ranking both more accurate, but only when their content is chosen carefully, not just when a judgment is shortened. Reducing a judgment to a structured summary reversed which query type retrieved best and made LLM re-ranking help in every method tested, instead of barely half of them. But the choice of *which fields* to keep mattered just as much as summarising at all: a summary that states the legal principle behind each citation outperformed one that only lists citation names, and both outperformed a summary that dropped citations from the start.

Q3: What does using the legal issue, rather than the facts, contribute to case law retrieval? It helps, but only on structured summaries, and only as one half of the picture. On full judgment text, facts retrieve better than issues, since a short issue phrase cannot match a 20,000-word document. On structured summaries this reverses, and issues retrieve better across every method tested. Facts and issues also do not find the same relevant cases: each one finds ground-truth cases the other misses, so using both together, not either alone, is what gives full coverage.

Q4: Can RAG help LLMs manage lengthy legal documents and reduce hallucination? Yes, for hallucination; only partly for length. RAG with a clear prompt reduced hallucination in both phases. For document length, RAG worked well once judgments were split into multiple passes rather than summarised in one call; a single-pass summary handled short and medium judgments but left long ones incomplete, and the longest judgments in the corpus still failed to produce a usable summary at all.

Q5: How well can RAG generate structured, template-like summaries from a single query under zero-shot learning? Well within the model’s context window; not reliably beyond it. A single zero-shot query produced a good structured summary for any judgment that fit the context window in one pass. Judgments longer than this needed to be split into several passes with a follow-up repair step; without that extra step, statutes and precedents were frequently left out or malformed.

6.5 Limitations

Hyperparameters such as chunk size, the BM25/FAISS weighting, and reranking prompt were selected by observing performance on the same 50-query benchmark used to report headline results, consistent with standard practice for this class of system but without a fully independent held-out validation set.

The retrieval cutoff (top 100–150 documents) and final shortlist size (top 20–30) were fixed by prompt-length constraints and not otherwise tuned.

Phase 1’s Gemini Pro declined to process some violent or sexual-offence judgments, narrowing the corpus and excluding affected queries; Phase 2 avoids this with Llama 3 70B but relies on a single LLM family throughout, so the reported MAP/MRR should not be read as an upper bound for other or larger models. Results are single-run point estimates and carry the run-to-run variance documented in Experiments 2,

3, and 6. Run-to-run variance was assessed on only 10 of the 50 queries due to budget constraints.

Four of 500 source judgments (C71, C342, C406, C481) failed generation due to a combination of long-document merge conflicts and transient infrastructure faults — intermittent 502 errors from the inference endpoint and a malformed-response parsing bug; individual-file retries later succeeded, but the available API quota was exhausted before the corpus could be regenerated and experiments rerun. Since C71 is ground truth for Q28 and Q37, both queries retain an unrecoverable one-case recall deficit in the reported results.

This work is also scoped to Indian case law, where judgments are largely unstructured and inconsistently formatted — the very condition that motivates the structured-summary approach. Whether the same gains hold for jurisdictions with more standardised judgment formats was not tested and is left for future work.

More fundamentally, even exhaustive human validation cannot fully eliminate residual error in LLM-generated content: the model may hallucinate legal principles, misattribute holdings to the wrong case, or omit material facts in ways that are subtle enough to pass expert review. This irreducible uncertainty is an inherent property of using a general-purpose LLM for zero-shot structured extraction from complex legal documents, and users of the corpus or any downstream system built on it should treat the structured summaries as high-quality approximations rather than authoritative legal records.

The structured summaries used in this study are LLM-generated and have not been exhaustively validated; while Phase 1 included manual expert review of 50 summaries, no manual validation was performed on the 496 summaries used in Phase 2. This is a significant limitation, as errors in the automatically generated summaries — such as hallucinated legal principles, misattributed holdings, or omitted facts — may have propagated undetected into the retrieval and reranking results.

None of these limitations preclude near-term use: the pipeline needs no fine-tuning and runs on modest hardware, making it usable today by individual practitioners or small chambers, while model choice, run stability, and cross-jurisdiction generalisability remain open.

7 Conclusion

This work built a system for retrieving Indian case law using a large language model, with no fine tuning. Gemini Pro reached a MAP of 0.365 on 50 judgments; Llama 3 70B reached about 0.31 on the full 496-judgment corpus under a harder, ranking aware metric. Structured summaries make this possible: shrinking a judgment to its key fields fits more cases into one LLM call, and which fields are kept matters, since a field stating the legal principle behind each citation beat both a citation-free format and one listing citation names alone. RAG cut hallucination with a clear prompt and summarised most judgments in one pass; only the longest needed several passes, and a few could not be summarised at all.

On facts versus issues, the two find different cases, so combining them gives the widest coverage. But mixing them dilutes the ranking: the issue-only pool ranked

correct cases higher than the combined pool did. This matches how lawyers search, by issue first, and the issue query alone was enough for both retrieval and re-ranking. A fuller facts query can help re-ranking, but a poorly written one can make the model ignore correct answers already in its context.

Two lessons sit above these results. At retrieval, the query needs to match the document in length and detail. At re-ranking, what matters is clarity, not length. Above both, how the document is represented matters more than the query.

Results are not fully deterministic and come from a single model per phase, so the comparison across phases is not a clean test of model choice alone. A few of the longest judgments could not be summarised, capping recall on queries that depend on them. Differences between configurations should be read as rough, not exact.

The structured summary is the most reusable part of this work, useful beyond case law retrieval alone for tasks such as statute retrieval, argument generation, judgment prediction, and knowledge graph construction. Since the method needs no fine-tuning and runs on modest hardware, it is usable today by individual lawyers and small firms, including with open-access models, at minimal cost. A natural use case for Indian law is a shared vector database of structured summaries for landmark judgments in a given practice area – for example, a lawyer handling income tax cases could build one from landmark income-tax judgments and use it both for this retrieval pipeline and for argument generation from the same set of structured summaries.

Four directions remain open. First, close the recall ceiling: cases that share a legal principle but no surface vocabulary are still often missed, and domain-adapted legal embeddings or broader pooling strategies are the likely fix. Second, find the right query length for a given document length, and build a better re-ranker for the union pool, since the cross-encoder we tried failed even though the pool already contains most of the relevant cases. Third, make LLM-generated queries more consistent, since results currently vary by about ± 0.03 to 0.04 MAP between runs of the same setup. Fourth, build better evaluation methods for this kind of generative legal output, since standard metrics like ROUGE do not fit it well, and expert review, while useful, does not scale.

Declarations

Funding. This work received no direct financial funding. Computational resources were provided through the IBM Watsonx Solution Grant awarded to support research, which enabled access to IBM Watsonx.ai and its foundational models used in this study.

Competing Interests. The authors received computational resource support through the IBM Watsonx Solution Grant for research purposes. IBM had no role in the study’s design, data collection, analysis, interpretation of the results, manuscript preparation, or the decision to publish. The authors declare no other competing interests.

Acknowledgement. The authors gratefully acknowledge the IBM Watsonx Solution Grant for providing the computational resources that enabled this research. We would like to note that this work was enabled by computational resources provided through

the IBM Watsonx Solution Grant, a selectively awarded global research support program. The grant provided access to IBM Watsonx.ai and foundational models but had no role in the scientific design, analysis, interpretation or reporting of this work.

Ethics approval. This study did not require ethics approval, as it involved only publicly available judgments and voluntary, anonymous expert review.

Consent to participate. The legal research scholars, Lawyers and law students who took part in the structured-summary and case-retrieval evaluation did so voluntarily and were informed of the purpose of the assessment.

Consent for publication. Not applicable. No identifiable participant data is included in this paper; all reviewers and scholars are referred to anonymously.

Data availability. The FIRE 2019 AILA judgment corpus is available from the dataset organisers cited in this paper [Bhattacharya et al. \(2019\)](#).

Supplementary information. The sample results of the AQgR for Case Law Retrieval is enclosed in the Appendix Section.

The sample outputs generated by the proposed method is included in this section. The output for Query 1 and Query 2 is included. The change in the output patterns can also be seen from the sample. With same prompt, the output generated is in different format. The sample output generated by structured summary generation is also included.

Appendix A Sample Output

A.1 Sample Output : Case Law Retrieval using Gemini

Q1—————*In 1961, the appellant was appointed as an officer in a bank and later promoted to Foreign Exchange Department. In 1964, he became involved in a society formed for employee housing. In 1969, an inquiry found him negligent in handling society funds, resulting in a loss of Rs. 3,59,000. The bank suspended him in 1970 after criminal charges were filed against him. The inquiry officer later found him liable for Rs. 2,36,000 to the society. The bank terminated his services in 1971 based on this finding. The appellant successfully appealed the criminal conviction in 1973. He filed a writ petition in 1975 for reinstatement, which was granted by the High Court in 1979. However, the Division Bench of the High Court reversed this decision in 1985 on grounds of laches and merits.*

1. Whether the Bank's termination of the appellant's services was legal and justified.
2. Whether the appellant was denied natural justice in the termination process.
3. Whether the High Court erred in dismissing the appellant's writ petition on grounds of laches and merits.

[...] response Relevant Judgments:

1. **Central Inland Water Transport Corporation Limited v. Brojo Nath Ganguly and Another** ◦ Relevance Score: 9 ◦ Legal Issue: Whether a service rule allowing termination of permanent employees without reasons is arbitrary and violative of Article 14 of the Constitution. ◦ Reason: The judgment struck down a similar service rule as void and directed reinstatement with back pay.

2. **West Bengal State Electricity Board and Others v. Desh Bandhu Ghosh and Others** ◦ Relevance Score: 8 ◦ Legal Issue: Whether a regulation allowing termination of permanent employee services without reasons is arbitrary and violative of Art. 14 of the Constitution. ◦ Reason: The judgment held that such a regulation is arbitrary and violates Article 14.
3. **O.P. Bhandari v. Indian Tourism Development Corporation Limited** ◦ Relevance Score: 7 ◦ Legal Issue: Whether a rule authorizing termination of an employee by giving notice or payment of salary in lieu thereof is constitutional. ◦ Reason: The judgment held that such a rule is unconstitutional and violates Articles 14 and 16 (1) of the Constitution.
4. **Ranchhodji Chaturji Thakore v. The Superintendent Engineer, Gujarat Electricity Board, Himmatnagar** ◦ Relevance Score: 6 ◦ Legal Issue: Whether an employee acquitted of criminal charges is entitled to reinstatement and back wages. ◦ Reason: The judgment held that reinstatement should be granted but back wages could be denied for the period the employee was not in service due to conviction.
5. **Jaipal Singh v. Union of India and Others** ◦ Relevance Score: 5 ◦ Legal Issue: Whether an employee acquitted of criminal charges is entitled to reinstatement and back wages. ◦ Reason: The judgment held that reinstatement should be granted but back wages could be denied for the period the employee was not in service due to conviction.

Q2—*The appellant, a former Cabinet Minister, testified as a witness in a bribery case against the first respondent, who was convicted. The High Court overturned the conviction, criticizing the appellant's conduct and credibility. The appellant had laid a trap for the respondent, who was arrested with marked currency in his briefcase. The prosecution claimed the respondent had demanded bribes monthly, while the respondent claimed the money was for donations. The High Court accepted the respondent's explanation and acquitted him, prompting the appellant to resign from his post to uphold democratic principles.*

1. Whether the disparaging remarks made by the Appellate Judge against the appellant, who was examined as prime witness in the trial and had become a Cabinet Minister at the time of the appeal, were justified and appropriate.
2. Whether the High Court erred in acquitting the first respondent based on its finding that the prosecution had not proved its case by acceptable evidence and that the first respondent's explanation for the possession of the currency notes appeared probable.
3. Whether the High Court's adverse remarks about the conduct of the appellant and the credibility of his testimony were supported by the evidence and whether they were fair and impartial.

[...] response

1. Whether the disparaging remarks made by the Appellate Judge against the appellant, who was examined as prime witness in the trial and had become a Cabinet Minister at the time of the appeal, were justified and appropriate. Relevance Score: 10 • Case Law: **R.K. Lakshmanan v. A.K. Srinivasan and Another** ◦ Reasoning: The Court held that there was ample material before the High

Court bearing on the impugned conduct of the appellant, justifying the adverse comments in question.

2. Whether the High Court erred in acquitting the first respondent based on its finding that the prosecution had not proved its case by acceptable evidence and that the first respondent's explanation for the possession of the currency notes appeared probable. Relevance Score: 8 • Case Law: **Mohammad Naim v. State of Uttar Pradesh** ◦ Reasoning: The High Court has inherent power to expunge its own remarks to prevent abuse of process or secure justice.
3. Whether the High Court's adverse remarks about the conduct of the appellant and the credibility of his testimony were supported by the evidence and whether they were fair and impartial. Relevance Score: 7 • Case Law: **K.R. Srinivas v. Dr R.M. Premchand** ◦ Reasoning: The Supreme Court held that Dr Premchand's writ petition was not maintainable as he had approached the High Court at a belated stage when the answer books had been destroyed, preventing effective cross-checking.

A.2 Sample Output: Structured Summary Generation

Structured Summary of Judgment

Appendix B Structured Summary of Judgments: Validation Questionnaire

Instructions:

I will furnish you with a judgment along with its automatically generated structured summary. Your task will be to verify the precision of the structured summary. You'll need to validate each primary field of the summary. The original judgment (e.g., C1.txt) and its corresponding summary (C1.json) are located in a folder named C1. To examine the contents in a structured format, you can utilize the website <https://codebeautify.org/jsonviewer>. Simply paste all the contents from the JSON file provided to you on the website. This will present you with a structured format, facilitating easy validation.

* Indicates required question

1. Email *
2. File name of the Judgments validated *

1. C1.txt
2. C9.txt
3. C14.txt
4. C21.txt
5. C22.txt
6. C25.txt
7. C27.txt

8. C31.txt
9. C38.txt
10. C47.txt
11. C49.txt
12. C54.txt
13. C59.txt
14. C65.txt
15. C69.txt
16. C72.txt
17. C75.txt
18. C76.txt
19. C79.txt
20. C82.txt
21. C85.txt
22. C92.txt
23. C93.txt
24. C94.txt
25. C121.txt
26. C122.txt
27. C126.txt
28. C139.txt
29. C141.txt
30. C147.txt
31. C152.txt
32. C155.txt
33. C162.txt
34. C164.txt
35. C170.txt
36. C171.txt
37. C182.txt
38. C184.txt
39. C186.txt
40. C2796.txt
41. C2797.txt
42. C2798.txt
43. C2799.txt
44. C2800.txt
45. C2801.txt
46. C2802.txt
47. C2803.txt
48. C2804.txt
49. C2805.txt
50. C2806.txt

3. Name

4. Designation (Law student/Legal Professional) *

5. Court *

- The system correctly identifies court name mentioned in the document and extracted it.
- The system accurately determines that there's no court mentioned in the document.
- The system incorrectly identifies court name. (Extracting wrong information)
- The court name is present in the document but the system could not extract it.

6. Case No *

- The system correctly identifies the case number mentioned in the document.
- The system accurately determines that there's no case number mentioned in the document.
- The system incorrectly identifies a case number as being present. (Extracting wrong information)
- The case number is present in the document, but the system could not extract it.

7. Kind of Judgment (Appeal/Petition) *

- The system correctly identifies the kind of judgment mentioned in the document.
- The system accurately determines that there's no kind of judgment mentioned in the document.
- The system incorrectly identifies a kind of judgment as being present. (Extracting wrong information)
- The kind of judgment is present in the document, but the system could not extract it.

8. Party Names *

- The system correctly identifies the parties mentioned in the document.
- The system accurately determines that there are no parties mentioned in the document.
- The system incorrectly identifies parties as being present. (Extracting wrong information)
- The parties are mentioned in the document, but the system could not extract them.

9. Date *

- The system correctly identifies the date mentioned in the document.
- The system accurately determines that there's no date mentioned in the document.

- The system incorrectly identifies a date as being present.
- The date is present in the document, but the system could not extract it.

21. Overall Quality of the summary *

- Very Satisfied
- Satisfied
- Neutral
- Dissatisfied
- Very Dissatisfied

22. Comments or Additional Features Needed in Future *

References

- Ashley, K.D.: Case-based reasoning and its implications for legal expert systems. *Artificial Intelligence and Law* (1992) <https://doi.org/10.1007/BF00114920>
- Askari, A., Verberne, S.: Combining lexical and neural retrieval with longformer-based summarization for effective case law retrieval. In: *CEUR Workshop Proceedings*, vol. 2848, pp. 8–19 (2021). <http://ceur-ws.org/Vol-2848/paper1.pdf>
- Bhattacharya, P., Ghosh, K., Majumder, P.: Fire 2019 aila track: Artificial intelligence for legal assistance. In: *ACM International Conference Proceeding Series*, pp. 25–32. ACM, ??? (2019). <https://doi.org/10.1145/3368567.3368587>
- Bhattacharya, P., Mehta, P., Ghosh, K., Ghosh, S., Pal, A., Bhattacharya, A., Majumder, P.: Fire 2020 aila track: Artificial intelligence for legal assistance. *ACM International Conference Proceeding Series*, 1–3 (2020) <https://doi.org/10.1145/3441501.3441510>
- Bhattacharya, P., Paul, S., Ghosh, K., Ghosh, S., Wyner, A.: Identification of rhetorical roles of sentences in indian legal judgments. *Frontiers in Artificial Intelligence and Applications* **322**, 3–12 (2019) <https://doi.org/10.3233/FAIA190301>
- Cormack, G.V., Clarke, C.L., Buettcher, S.: Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In: *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 758–759 (2009)
- D’Agostini Bueno, T.C., Wangenheim, C., Barcia, R.M.: Jurisconsulto: Retrieval in jurisprudencial text bases using juridical terminology. In: *Proceedings of the International Conference on Artificial Intelligence and Law*, pp. 22–29 (1999). <https://doi.org/10.1145/316019.316024> . ACM
- Gupta, A., Furniturewala, S., Kumari, V., Sharma, Y.: Steno ai at semeval-2023 task

- 6: Rhetorical role labeling of legal documents using transformers and graph neural networks. 17th International Workshop on Semantic Evaluation, SemEval 2023 - Proceedings of the Workshop, 1858–1862 (2023) <https://doi.org/10.18653/v1/2023.semeval-1.256>
- Han, J., Burgess, P., Shareghi, E.: Legal citation prediction with llms: a comparative evaluation of instruction tuning, retrieval, and jurisdiction-specific pre-training on the auslaw citation benchmark. *Artificial Intelligence and Law*, 1–35 (2026)
- Kim, M.-Y., Rabelo, J., Goebel, R.: Statute law information retrieval and entailment. In: *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*, pp. 283–289 (2019)
- Kim, M.Y., Rabelo, J., Satoh, K.: Coliee 2022 summary: Methods for legal document retrieval and entailment. In: *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 13956, pp. 3–19. Springer, ??? (2023). https://doi.org/10.1007/978-3-031-29168-5_4
- Klein, M.C., Van Steenbergen, W., Van Harmelen, F.: Thesaurus-based retrieval of case law. In: *Frontiers in Artificial Intelligence and Applications*, vol. 133, pp. 35–42. IOS Press, ??? (2006). <https://doi.org/10.3233/978-1-58603-952-3-63>
- Leburu-Dingalo, T., Motlogelwa, N.P., Modungo, M.: Ub at fire 2020 precedent and statute retrieval. In: *CEUR Workshop Proceedings*, vol. 2782, pp. 103–106 (2020). <http://ceur-ws.org/Vol-2782/ub.pdf>
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.T., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-augmented generation for knowledge-intensive nlp tasks, vol. 2020-December (2020)
- Mandal, A., Ghosh, K., Mandal, S.: Unsupervised approaches for measuring textual similarity between legal court case reports. *Artificial Intelligence and Law* (2021) <https://doi.org/10.1007/s10506-020-09280-2>
- Maxwell, K.T., Schafer, B.: Concept and context in legal information retrieval. In: *Frontiers in Artificial Intelligence and Applications*, vol. 185, pp. 249–254. IOS Press, ??? (2008). <https://doi.org/10.3233/978-1-58603-952-3-63>
- Nigam, S.K., Dubey, T., Shallum, N., Bhattacharya, A.: Segment first, retrieve better: Realistic legal search via rhetorical role-based queries. *arXiv preprint arXiv:2508.00679* (2025)
- Noviello, Y., Pallotta, E., Pinzarrone, F., Tanzi, G.: Teamunibo at semeval-2023 task 6: A transformer based approach to rhetorical roles prediction and ner in legal texts. 17th International Workshop on Semantic Evaluation, SemEval 2023 - Proceedings of the Workshop, 275–284 (2023) <https://doi.org/10.18653/v1/2023.semeval-1.37>

- Nguyen, H.T., Phi, M.K., Tu, M.P.: Attentive deep neural networks for legal document retrieval. *Artificial Intelligence and Law* (2022) <https://doi.org/10.1007/s10506-022-09341-8>
- Paul, S., Ghumare, D., Goyal, P., Ghosh, S., Modi, A.: Il-pcsr: Legal corpus for prior case and statute retrieval. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 14599–14622 (2025)
- Pandian, S., Joshi, S.: Autosuggestion of relevant cases and statutes. In: *Frontiers in Artificial Intelligence and Applications* vol. 343, pp. 129–136. IOS Press, ??? (2022). <https://doi.org/10.3233/FAIA220471>
- Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al.: Improving language understanding by generative pre-training (2018)
- Savelka, J., Ashley, K.D., Gray, M.A., Westermann, H., Xu, H.: Explaining legal concepts with augmented large language models (gpt-4) (2023)
- Sampath, K., Durairaj, T.: Prelcap : Precedence retrieval from legal documents using catch phrases. *Neural Processing Letters* **54**, 3873–3891 (2022) <https://doi.org/10.1007/s11063-022-10791-z>
- Santosh, T., Nolasco, I.M.O., Grabmair, M.: Lecopcr: Legal concept-guided prior case retrieval for european court of human rights cases. In: *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1654–1661 (2025)
- Sun, Z., Zhang, K., Yu, W., Wang, H., Xu, J.: Logic rules as explanations for legal case retrieval. In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 10747–10759 (2024)
- Vuong, Y.T.H., Bui, Q.M., Nguyen, L.M.: Sm-bert-cr: a deep learning approach for case law retrieval with supporting model. *Artificial Intelligence and Law* (2023) <https://doi.org/10.1007/s10506-022-09319-6>
- Opijnen, M., Santos, C.: On the concept of relevance in legal information retrieval. *Artificial Intelligence and Law* (2017) <https://doi.org/10.1007/s10506-017-9195-8>
- Vats, S., Zope, A., De, S., Sharma, A., Bhattacharya, U., Nigam, S.K., Guha, S.K., Rudra, K., Ghosh, K.: Llms—the good, the bad or the indispensable?: A use case on legal statute prediction and legal judgment prediction on indian court cases. In: *The 2023 Conference on Empirical Methods in Natural Language Processing* (2023)
- Winning Strategies for Court Cases in India: Essential Factors and Effective Techniques. <https://www.centurylawfirm.in/blog/winning-strategies-for-court-cases-in-india-essential-factors-and-effective-techniques/>,

- Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E.H., Le, Q.V., Zhou, D.: Chain-of-thought prompting elicits reasoning in large language models. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. NIPS '22. Curran Associates Inc., Red Hook, NY, USA (2022)
- Yu, F., Quartey, L., Schilder, F.: Exploring the effectiveness of prompt engineering for legal reasoning tasks. Proceedings of the Annual Meeting of the Association for Computational Linguistics, 13582–13596 (2023) <https://doi.org/10.18653/v1/2023.findings-acl.858>
- Yu, W., Sun, Z., Xu, J., Dong, Z., Chen, X., Xu, H., Wen, J.-R.: Explainable legal case matching via inverse optimal transport-based rationale extraction. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 657–668 (2022)
- Zheng, L., Guha, N., Arifov, J., Zhang, S., Skreta, M., Manning, C.D., Henderson, P., Ho, D.E.: A reasoning-focused legal retrieval benchmark. In: Proceedings of the 2025 Symposium on Computer Science and Law, pp. 169–193 (2025)

Key	Value
Court	Supreme Court of India
Case No	Civil Appeal No. 562 of 1985
Kind of Judgment	Appeal
Parties	• Appellant: West Bengal State Electricity Board and Others • Respondent: Desh Bandhu Ghosh and Others
Date	26 February 1985
Bench of Judges	O. Chinnappa Reddy, J.
Facts	Termination of respondent's services (a permanent employee) without reasons under Regulation 34 of the Board's regulations, allowing termination with three months' notice or salary in lieu.
Arguments	Appellant's Arguments: • Regulation 34 does not offend Article 14 of the Constitution. • Sections 18A and 19 of the Electricity Supply Act provide sufficient guidelines. • Power to terminate services vested in higher-ranking officials, likely to be exercised reasonably. Respondent's Arguments: • Regulation 34 is arbitrary and enables discrimination. • The rule is a "hire and fire" policy, outdated and should be abolished.
Issues or Questions	Whether Regulation 34 of the Board's regulations, allowing termination of permanent employee services without reasons, is arbitrary and violative of Article 14 of the Constitution.
Reasoning	• Regulation 34 is arbitrary and confers a power capable of vicious discrimination. • It is a "hire and fire" rule with no guidelines or limitations. • Similar rules have been struck down by this Court as violative of Article 14.
Case Disposed in Favor of	Respondent
Final Judgment	Appeal dismissed with costs.
Statutes Referred	Electricity Supply Act: • Principle: Guidelines for termination of services. • Application: Sections 18A and 19 provide some guidelines, but not sufficient to save Regulation 34 from being arbitrary.
Precedents Referred	Moti Ram Deka v. North East Frontier Railway: • Principle: Rules allowing termination without inquiry or reasons may be contrary to Article 311(2) and Article 14 of the Constitution. • Application: Cited as an example of a rule struck down for being arbitrary and violative of Article 14. S. S. Muley v. J.R.D. Tata and Ors.: • Principle: Standing Order allowing dismissal without inquiry or reasons is violative of natural justice. • Application: Cited as an example of a rule struck down for being arbitrary and violating principles of natural justice. Workman, Hindustan Steel Ltd. v. Hindustan Steel Ltd.: • Principle: Standing Order allowing dismissal without inquiry or reasons violates natural justice. • Application: Cited as an example of a rule struck down for being arbitrary and violating natural justice. Manohar P. Kharkhar v. Raghuraj: • Principle: Complexities of modern administration may necessitate powers like those under Regulation 48. • Application: Cited by the appellant to support Regulation 34, but the Court disagrees with its reasoning.
New Legal Principles	Arbitrary and uncensored powers of termination of employment, without reasons or inquiry, are violative of Article 14 of the Constitution and natural justice; This principle can be applied to future cases involving challenges to termination rules or regulations.
Important Subjects Discussed	Arbitrary Power; Discrimination; Natural Justice; Termination of Employment.

Table A1 Structured Summary of Judgment