# KRISHNA PRAKASH

## Krishna_Thesis

Thesis check Krishna

### Document Details

**Submission ID**
trn:oid:::27535:139739359

**Submission Date**
May 20, 2026, 6:14 PM GMT+5:30

**Download Date**
May 20, 2026, 6:22 PM GMT+5:30

**File Name**
Krishna_Thesis.pdf

**File Size**
4.5 MB

**37 Pages**

**15,430 Words**

**84,453 Characters**

# 3% Overall Similarity

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

## Filtered from the Report

- Bibliography
- Small Matches (less than 8 words)

## Match Groups

**28** Not Cited or Quoted **2%**
Matches with neither in-text citation nor quotation marks

**16** Missing Quotations **1%**
Matches that are still very similar to source material

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

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

## Top Sources

1% Internet sources
1% Publications
2% Submitted works (Student Papers)

## Match Groups

- **28** Not Cited or Quoted 2%
  Matches with neither in-text citation nor quotation marks
- **16** Missing Quotations 1%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

## Top Sources

1% Internet sources
1% Publications
2% Submitted works (Student Papers)

## Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

| | | |
|---|---|---|
| 1 | Student papers | University of Hull on 2026-05-08 <1% |
| 2 | Internet | arxiv.org <1% |
| 3 | Internet | www.arxiv-vanity.com <1% |
| 4 | Student papers | Vellore Institute of Technology on 2026-04-29 <1% |
| 5 | Student papers | University of St Andrews on 2026-04-07 <1% |
| 6 | Student papers | Queen Mary and Westfield College on 2026-03-05 <1% |
| 7 | Internet | aclanthology.org <1% |
| 8 | Student papers | International Institute of Information Technology, Hyderabad on 2026-05-05 <1% |
| 9 | Student papers | King's College on 2026-04-23 <1% |
| 10 | Student papers | Liverpool John Moores University on 2026-04-29 <1% |

11 Publication

Mohammad Aqib, Mohd Hamza, Qipei Mei, Ying Hei Chui. "Fine-tuning large lang... <1%

12 Student papers

Queen's University of Belfast on 2024-04-10 <1%

13 Student papers

University of Sheffield on 2024-04-19 <1%

14 Internet

hal.science <1%

15 Publication

"Advances in Information Retrieval", Springer Science and Business Media LLC, 20... <1%

16 Student papers

The University of Manchester on 2025-08-30 <1%

17 Student papers

University of Hertfordshire on 2026-04-26 <1%

18 Student papers

University of Leicester LTI on 2026-04-27 <1%

19 Student papers

University of Ulster on 2026-04-19 <1%

20 Student papers

University of Ulster on 2026-05-04 <1%

21 Internet

ir.iit.edu <1%

22 Internet

pingcap.github.io <1%

23 Internet

rucore.libraries.rutgers.edu <1%

24 Student papers

Angelo State University (Blackboard LTI 1.3, Prod) on 2026-05-18 <1%

25 Student papers

GL Bajaj Institute of Technology and Management on 2026-05-04 <1%

26 Student papers

University of Queensland on 2026-04-09 <1%

27 Student papers

University of Ulster on 2026-05-01 <1%

28 Student papers

University of York on 2025-04-30 <1%

29 Student papers

Vellore Institute of Technology on 2026-04-22 <1%

30 Internet

elibrary-dev.nusamandiri.ac.id <1%

31 Internet

files01.core.ac.uk <1%

32 Internet

www.dei.unipd.it <1%

# ABSTRACT

Many systems built around large language models have to answer questions whose answers the model itself does not hold. The answer sits in an external collection of documents, and a retrieval step is the only way to bring that content into the model's context. Retrieval-augmented generation (RAG) is a standard architecture for this setting, but the default version of it was designed for free-form web text. Its accuracy degrades on collections that are hierarchical, that are revised over time, and that demand exact citations.

This thesis develops a framework for such collections in two parts. The first part is a structural pipeline that respects the hierarchy of the source documents at every stage. It splits text at section and clause boundaries, attaches citation-grade metadata to each chunk, runs a dense neural retriever and a sparse term-matching retriever in parallel, fuses their ranked lists, reranks with a cross-encoder that uses the metadata as an extra signal, and constrains the language model to a fixed output format. The second part is a stateful agentic layer that wraps a lightweight controller around the structural pipeline. The controller maintains a working memory across retrieval calls, reflects on the evidence collected so far, decides whether the evidence is sufficient, and issues a refined query when it is not.

The structural pipeline is implemented and evaluated on a corpus of central Indian statutes against a benchmark of two hundred test queries whose ground-truth applicable provisions were prepared by a practising domain expert. The pipeline identifies the primary applicable citation in 94.0% of test queries (188 of 200; 95% Wilson interval [89.8%, 96.5%]) and returns the full applicable citation set in 82.0% (164 of 200; 95% Wilson interval [76.1%, 86.7%]). Every neural component is used without domain-specific fine-tuning, so the reported numbers reflect the pipeline's behaviour in the absence of any domain-adaptation effect. The stateful agentic controller is presented as a fully specified design, composed from methods already published in the agentic and self-reflective retrieval literature; its end-to-end empirical evaluation is left for subsequent work.

# CHAPTER 1

# INTRODUCTION

A language model trained on the open web can write fluent prose, answer general-knowledge questions, and follow instructions. It cannot answer questions whose answers are absent from its training data, and it cannot retain anything beyond its context window. Both limits become consequential when the model is asked to reason about a body of authoritative text that is too large to fit in one prompt and too specialised to be present in the training corpus: for example, a body of statutes, a set of clinical guidelines, a regulatory standard, or a body of scientific literature. The architecture that brings external text into a language model's context at inference time is retrieval-augmented generation (RAG) [1]; the chapters that follow specify the form RAG must take before it can support reasoning in such settings.

## 1.1 Reasoning Over External Knowledge

Two properties of contemporary language models motivate the problem addressed in this thesis. The first is that the parametric knowledge of the model is fixed at the moment of training. Any amendment, correction, or new edition of the source after that moment is invisible to the model unless an external mechanism supplies it. The in-context retrieval-augmented setting [2] is the established way to inject such updates at inference time without retraining. The second property is that even when the relevant information is brought inside the context window, the model does not use it uniformly: performance follows a U-shaped curve, peaking at the start and the end of the input and dropping in the middle [3]. Adding more context, on its own, does not produce more reasoning.

A different way to state the same observation is that contextual reasoning is a separate problem from contextual retrieval. The retrieval problem is to place the relevant text inside the model's context window. The reasoning problem is to compose an answer from that text. The two interact: a missed retrieval guarantees a wrong answer, but a successful retrieval does not guarantee a correct one. A retrieval-augmented system intended for knowledge-intensive applications must treat both problems as first-class concerns, not only the first.

### 1.1.1 Knowledge-Intensive Corpora

The settings in which both problems become acute are *knowledge-intensive*: applications whose answers must be grounded in a body of authoritative text that the model has not memorised. Four properties recur across the corpora considered in this thesis and define the operational notion of *knowledge-intensive* used throughout the document.

(i) **Large size**, well beyond what fits inside any single language-model context window.

(ii) **Hierarchical structure**, where the citable unit is a section, clause, recommendation, or paragraph rather than the whole document.

(iii) **Periodic revision**, where newer versions of a provision can supersede older ones while the corpus retains both.

(iv) **Strict citation grounding**, where any claim drawn from the corpus has to be tied to a

precise location in it.

For the rest of the thesis, the strict reading of "knowledge-intensive" refers to a corpus that exhibits all four properties simultaneously; the empirical evaluation in Chapter 4 uses one such corpus. Elsewhere in the document the looser adjectives "specialist", "structured", or "citation-grounded" are used when only a subset of the four properties is relevant to the argument being made.

## 1.2 Failure Modes of the Default Pipeline

The version of RAG most commonly described in reference implementations and introductory accounts follows a single template: split each document into fixed-token windows, embed each window with a single dense encoder, fetch the top-$k$ windows by inner product at query time, and pass them to a generator. This default pipeline issues exactly one retrieval call per query. Three properties of this pipeline cause its failures on knowledge-intensive workloads, and the rest of the thesis is organised around them.

**Stateless retrieval.** A single retrieval call is the unit of work. If the call misses, the pipeline has no mechanism to recover. If the answer requires evidence from several parts of the corpus, the pipeline has no mechanism to compose them. Multi-step question answering exhibits the same failure mode: information that should have been retrieved at the first step is required only after the retriever's single call has already returned, leaving no opportunity for a second fetch [4]. Recent work on iterative and self-reflective retrieval [5, 6] attacks the same problem from the other side, treating retrieval as a step inside a longer reasoning loop rather than as a one-shot fetch. The ReAct formulation [7] makes this loop explicit: the model alternates between reasoning steps and external actions, and decides what to do next based on what it has just seen.

**Structural blindness in chunking.** A pipeline trained on free-form web text treats a document as a stream of tokens. Knowledge-intensive corpora are not streams. A statute is a list of sections, each of which is the indivisible citable unit. A clinical guideline is a list of recommendations. A standard is a list of clauses. When a fixed-size chunk begins inside one section and ends inside the next, the resulting passage cannot be used as a citation even if its embedding is close to the query, because the citation metadata is ambiguous. The boundary problem is the cause, not the chunk size.

**Encoder mismatch.** The dense and sparse retrievers used in a default pipeline are trained on general-purpose data. They do not encode the in-domain meaning of an act name or a section number, and they lose accuracy on specialist corpora compared to their in-distribution numbers [8, 9]. A sparse retriever [10] compensates on exact terms but reintroduces vocabulary mismatch in the other direction.

The three properties are independent. Resolving chunking does not resolve stateless retrieval, resolving stateless retrieval does not resolve the encoder mismatch, and resolving the encoder mismatch does not resolve chunking. A retrieval-augmented system intended for knowledge-intensive applications must address all three; the framework developed in the chapters that follow does so explicitly.

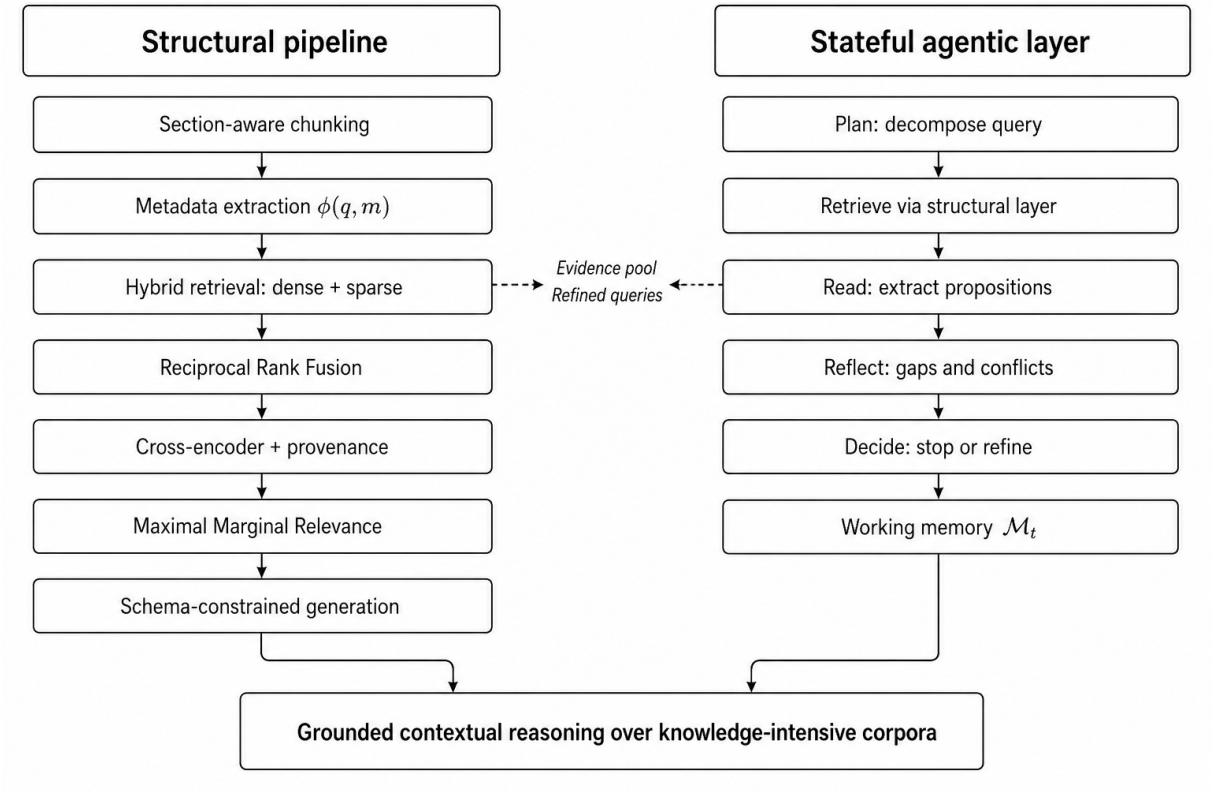


**Figure 1.1.** Two layers of the proposed framework.

## 1.3 Structure and State as Complementary Mechanisms

Two properties are absent from the default pipeline and are restored by the framework developed in this thesis: *structure* and *state*. Each addresses one half of the contextual reasoning problem.

**Structural context within a single retrieval call.** If chunks respect the citation grain of the corpus, the metadata that defines each citation can travel with each chunk into the retriever and the reranker. The retriever can then use the metadata as an additional signal, the generator can produce citations that survive an end-to-end check, and the pipeline can operate on a precise (document, section, clause) tuple rather than on a positional window.

**State across many retrieval calls.** If the retriever runs inside a loop that carries a working memory, the loop can recover from missed retrievals by issuing a refined query, compose evidence from several calls before producing an answer, and stop when the accumulated evidence is sufficient. This is the control structure followed by recent agentic and self-reflective retrieval methods, including ReAct [7], Self-RAG [6], FLARE [5], Self-Refine [11], and Reflexion [12].

The thesis develops both directions inside a single framework. The structural layer is a five-stage pipeline that respects document hierarchy at every stage: section-aware chunking, metadata extraction, hybrid retrieval combined through Reciprocal Rank Fusion (RRF) [13], cross-encoder reranking with a metadata provenance bonus, Maximal Marginal Relevance (MMR) [14] diversity selection, and schema-constrained generation. The stateful layer wraps a controller around this pipeline so that retrieval becomes one action inside a longer reasoning loop, with a persistent working memory, a reflect-and-refine step, and a confidence-gated stopping rule.

Figure 1.1 previews the two layers and their interface.

## 1.4 Contributions

The principal contributions of the thesis are the following.

1. A unified framework that combines a structural retrieval pipeline with a stateful agentic controller for contextual reasoning over knowledge-intensive corpora, with a narrowly defined interface (the evidence pool and the query stream) between the two layers.

2. A structural pipeline that respects citation grain at every stage. Its components are: priority-ordered, section-aware chunking with a fallback hierarchy; regular expression metadata extraction; rank-level fusion of dense and sparse candidates; cross-encoder reranking with an additive metadata provenance bonus; diversity selection; and schema-constrained generation. The pipeline is implemented and evaluated end-to-end.

3. A stateful agentic controller for the same framework, with a persistent working memory, an explicit plan-retrieve-read-reflect-decide loop, and a confidence-gated stopping rule. The controller is composed from methods already established in the agentic and self-reflective retrieval literature; it is presented in this thesis as a design-only contribution, and its end-to-end empirical evaluation against the same benchmark is reserved for subsequent work (Section 6.3).

4. An end-to-end empirical evaluation of the structural pipeline on a corpus chosen as an instantiation of the four knowledge-intensive properties of Section 1.1.1. The evaluation is positioned as evidence about the framework's behaviour: it quantifies the gap between primary-citation accuracy and full-citation-set accuracy and traces the residual error to the failure modes that the stateful controller is designed to address.

## 1.5 Scope

The thesis evaluates the structural layer end-to-end and presents the stateful agentic layer as a fully specified design. The end-to-end empirical evaluation of the stateful layer against the same benchmark is left to subsequent work. All neural components used in the empirical evaluation are off-the-shelf, so the reported numbers do not include any benefit that domain fine-tuning of the encoder or reranker might add; component-level attribution is reserved for the future ablation study identified in Section 6.3. The corpus used in the evaluation is in English, and the cross-lingual case is identified as future work.

Chapter 2 situates the framework against prior work on retrieval-augmented generation, agentic and self-reflective retrieval, structure-preserving retrieval, and knowledge-intensive domains, and articulates the gap in the literature that the thesis addresses.

# CHAPTER 2

# RELATED WORK

Agentic, stateful retrieval and structure-preserving retrieval have grown into two largely independent lines of work. Each of them answers part of the problem set out in Chapter 1, but neither answers all of it. This chapter argues that the two lines must be combined rather than treated as alternatives, and that the combination is what a knowledge-intensive retrieval-augmented system requires.

## 2.1 Retrieval-Augmented Generation

The neural backbone shared by all retrieval-augmented systems is the transformer [15], applied first to encoder pre-training [16], then to encoder-decoder pre-training [17], and to large decoder-only language models [18]. The RAG architecture combines a retriever and a generator inside a single answering pipeline. The original RAG system [1] paired a dense retriever with a generator in the Bidirectional and Auto-Regressive Transformers (BART) family [19] and trained the two end-to-end with an objective that marginalised over the choice of passage. REALM [20] introduced a related architecture for retrieval-augmented masked language modelling, and Fusion-in-Decoder [21] simplified the generator side by feeding all retrieved passages through a single decoder. RETRO [22] pushed the idea into the pre-training stage with a frozen retriever over a trillion-token corpus.

A subsequent body of work showed that modern decoder language models do not need to be trained jointly with the retriever. Prepending retrieved passages to the prompt at inference time recovers most of the benefit of a trained retrieval-augmented model [2]. This in-context formulation is the one the thesis adopts because it lets the generator be replaced without retraining the retriever.

In all of these systems the retriever is called once per query. The retrieved passages enter the generator's context window, and the generator produces an answer. There is no state between queries, no recovery mechanism if the retriever's single call misses, and no compositional procedure for combining evidence from separate calls. The default RAG pipeline is, in this sense, stateless.

## 2.2 Limitations of Stateless Retrieval

Recent work supports three claims about single-shot retrieval on a structured corpus.

**Multi-hop evidence requirements.** On multi-step question answering benchmarks, a single retrieval call frequently fails to surface every passage that the answer depends on. Interleaving retrieval with chain-of-thought reasoning, so that each step in the reasoning can issue its own retrieval call, recovers a large fraction of the missed evidence [4].

**Long context versus effective context.** A language model with a long context window does not use that window uniformly. The effect reported in [3] takes the form of a U-shaped accuracy

**Table 2.1.** Retrieval approaches and their trade-offs.

| Approach | Strength | Weakness | Cost |
|---|---|---|---|
| Boolean | exact match on indexed terms | recall loss on paraphrase | low |
| Okapi BM25 | weighted exact-term match using inverse document frequency | no semantic similarity | low |
| Dense retriever | semantic similarity in embedding space | rare-token miss | medium |
| Hybrid (dense+BM25) | combines exact and semantic signals | two indexes plus fusion parameter | medium |

curve: performance is highest when relevant information sits at the start or end of the context and lowest when it sits in the middle. Retrieving more passages and inserting them into the prompt does not solve the reasoning problem behind it; the model still fails to use the middle of its input. Reasoning is therefore not a function of retrieval bandwidth alone.

**Residual hallucination under retrieval grounding.** The introduction of retrieval grounding lowers the rate of unsupported claims in dialogue and summarisation [23, 24], but the rate never falls to zero. The survey of hallucination in natural language generation [25] catalogues the contributing factors across generation tasks and recommends retrieval grounding, faithfulness against the source, and consistency over generated text as the main mitigations. The structural and stateful layers of this thesis contribute to these mitigations at different stages of the pipeline.

Table 2.1 groups the retrieval approaches discussed in this and the following section. The grouping makes the design trade-offs explicit and helps locate the structural layer of the thesis inside it.

## 2.3 Agentic and Stateful Retrieval-Augmented Generation

The lines of work in this section share one move: they replace a single-shot retrieval call with a controller that maintains a state and issues several actions in sequence. The state may be the reasoning trace, the running draft of the answer, an episodic memory of past attempts, or a structured working memory. The actions may be retrieval calls, tool calls, or revisions of the output. The thesis composes a controller from these primitives.

**Externalised reasoning.** Chain-of-thought (CoT) prompting [26] gave language models a way to write their intermediate reasoning down as part of the output. The written reasoning trace is a convenient medium for a controller's state, and is a precondition for the agentic methods that follow.

**Action and observation loops.** The ReAct framework [7] interleaved reasoning steps with external actions (retrieval calls, tool calls) and observations returned by those actions, so that the model could decide what to do next based on what it had just seen. Toolformer [27] taught a language model to insert tool calls into its own output through self-supervised data construction.

**Self-reflection and self-refinement.** Self-Refine [11] and Reflexion [12] added an explicit reflection step in which the model criticises its own intermediate output and uses the critique to

**Table 2.2.** Agentic and iterative retrieval methods compared.

| Method | Step trigger | Step action | State carried |
|---|---|---|---|
| Chain-of-Thought [26] | every step | reason | reasoning trace |
| ReAct [7] | reasoning step | retrieve or call a tool | trace and observations |
| Self-Refine [11] | inspection of draft | rewrite the draft | last draft |
| Reflexion [12] | task failure | rewrite the policy | episodic memory |
| Self-RAG [6] | learned token | retrieve or suppress | emitted tokens |
| FLARE [5] | low token confidence | retrieve | partial draft |
| IRCoT [4] | each CoT step | retrieve | CoT path |
| HyDE [28] | initial query | draft answer, then retrieve | single-step draft |

drive a revision. The reflection produces an artefact (a critique, an episodic memory entry) that survives the next iteration.

**Retrieval-side gating.** Self-RAG [6] applied the reflection idea to retrieval itself: the model emits special control tokens that decide when to retrieve, when to stop, and when to suppress an unsupported claim. Active Retrieval Augmented Generation (FLARE) [5] reached a similar effect with a different signal, triggering a fresh retrieval whenever the generator's next-token probability dropped below a threshold. Hypothetical Document Embeddings (HyDE) [28] rewrite the query before retrieving by prompting the model to draft a hypothetical answer and embedding that draft.

**Persistent agent memory.** A separate line of work studies agents that maintain long-running memory across many actions [29]. The application domain there is different, but the architectural pattern (a controller that reads and writes a persistent memory) carries over to retrieval-augmented reasoning, and is the pattern adopted by the stateful layer of this thesis.

An interleaved retrieval and chain-of-thought (IRCoT) [4] controller runs a retrieval call at every step of the reasoning trace; it is the closest of these methods to the controller composed in this thesis, except that the stateful layer here writes to a more structured working memory and uses a confidence-gated stopping rule rather than a fixed retrieval schedule. Table 2.2 compares the methods above by what triggers a new step, what action the step takes, and what state survives between steps.

The methods above share two limitations, both individually and considered as a group. They take the structure of the retrieval index as given, and they are typically evaluated on benchmarks much smaller than a full statutory or regulatory corpus, where a single general-purpose dense encoder is sufficient. The stateful layer in this thesis composes these methods, but it composes them on top of a structural layer that treats the index as a design problem in its own right.

## 2.4 Structure-Preserving Retrieval

The retrieval side of a RAG pipeline has its own line of work, and this section reviews the parts the structural layer of the thesis composes.

**Sparse retrieval.** Boolean and probabilistic ranking methods scored documents by surface overlap with the query [30–33]. The Okapi Best-Matching-25 (BM25) family [10], which weights each query term by its inverse document frequency (IDF) and its within-document frequency, became the strongest single sparse retriever on most ad hoc retrieval collections and remains a competitive baseline against subsequent neural methods [34].

**Dense retrieval.** A sentence encoder maps any input string to a fixed-length vector in a shared embedding space, and similarity search returns the nearest neighbours of the query vector. Sentence-BERT [35] introduced the encoder used by the structural layer of this thesis. Dense Passage Retrieval [36] showed that a sentence encoder trained on labelled question-passage pairs outperforms BM25 on open-domain question answering. Later work added stronger negatives and curriculum sampling [37–39] or moved similarity scoring to the token level for finer matching [40].

**Zero-shot transfer to specialist corpora.** The Benchmarking Information Retrieval (BEIR) benchmark [8] measured zero-shot transfer of dense retrievers to eighteen out-of-domain collections, reporting results in normalised discounted cumulative gain at rank ten (nDCG@10). Dense retrievers trained on web-scale question answering lost noticeable accuracy on most out-of-domain collections, and on several specialist collections they failed to outperform BM25. A separate line of work in the legal domain showed that pre-training an encoder on in-domain text recovers part of that loss on downstream classification and retrieval [9]. The lesson the thesis takes from the two together is that no single retriever, dense or sparse, dominates on specialist corpora, and that a hybrid retriever is a reasonable response.

**Hybrid retrieval and rank fusion.** Combining dense and sparse retrievers requires reconciling two score scales. Reciprocal Rank Fusion [13] sidesteps this by working with ranks rather than scores. The smoothing constant $k = 60$ reported in [13] is the value reused in this thesis; it is unrelated to the BM25 parameters $k_1$ and $b$ in Equation 3.2.

**Neural reranking.** A two-stage architecture trades query-time computation for ranking precision: a first-stage retriever returns a small set of candidates and a cross-encoder rescores them with full attention between the query and each passage [41]. The cross-encoder is too expensive to run over the full corpus but inexpensive over a few dozen candidates. The treatment in [41] also discusses the empirical observation that cross-encoders trained on web-search queries lose accuracy on specialist corpora, which is again consistent with the BEIR results.

**Indexing.** The Facebook AI Similarity Search (FAISS) library [42] provides exact inner-product search for corpora up to roughly $10^5$ vectors, and Hierarchical Navigable Small World (HNSW) graphs [43] provide approximate search beyond that size. The structural layer of this thesis stays in the exact-search regime because the corpus used in the empirical evaluation is well inside it.

**Diversity selection.** Maximal Marginal Relevance [14] maximises a convex combination of relevance to the query and dissimilarity to already-selected passages. The structural layer of this thesis uses it as the final step before generation, so that the top-$k$ contexts do not all come from the same section of the corpus.

## 2.5 Knowledge-Intensive Domains

The corpora to which the framework is targeted share four recurring properties (Section 1.1.1): they are large, hierarchically structured, periodically revised, and judged by the precision of their citations. Several specialist domains exhibit all four properties.

**Legal text as a representative case.** Legal text exhibits all four properties in a direct form. Hierarchy is enforced by the act-section-clause structure of statutes; citation grain is enforced by the rules of legal writing; periodic revision is enforced by the legislative process; encoder mismatch is documented for legal corpora in particular, where a domain-trained encoder recovers part of the loss that a web-trained one incurs [9]. Earlier work in the area covers charge prediction, statute classification, case similarity and question answering [44–48]. A recent retrieval-augmented system for Indian statutory text [49] pairs a dense retriever with a generator on a small statutory corpus. The empirical evaluation in this thesis uses an Indian statutory corpus because it instantiates the four properties without further engineering, not because legal-question answering is the subject of the thesis.

**Other specialist corpora.** Beyond the legal domain, the BEIR benchmark contains zero-shot evaluations on biomedical fact verification, on scientific articles, and on financial question answering [8]. The pattern reported on those collections is qualitatively similar to the pattern reported in the legal-domain encoder work [9]: a dense retriever trained on web text loses accuracy on specialist collections, and a sparse baseline becomes competitive again. The methodological implication is the same in each of these domains. Treat the retriever as one signal among several, and respect the structure of the source corpus when building the index.

## 2.6 Positioning Relative to Prior Work

Three implications follow from the work reviewed above.

First, no single retriever, dense or sparse, dominates on hierarchically structured corpora, but a hybrid that combines the two through rank fusion is competitive and inherits the recall properties of both.

Second, a cross-encoder reranker is a low-cost way to recover precision after a first-stage retrieval, but a web-trained cross-encoder loses accuracy on specialist corpora and needs additional signal beyond the candidate text.

Third, the move from a one-shot retrieval call to a stateful controller has been studied in isolation from the structural concerns of the retrieval stage. The agentic methods of Section 2.3 take the retriever as given.

The thesis combines the implications of these three observations. On the structural side, it adds document-hierarchy chunking and extracted metadata that travel with each chunk into the reranker as an additional signal. On the stateful side, it composes the state, reflection, and re-querying primitives used by ReAct, Self-RAG, FLARE, Self-Refine, and Reflexion [5–7, 11, 12] into a single loop that runs on top of the structural layer. The central claim of the thesis is therefore not any individual component but the interface that makes structure and state composable.

Chapter 3 formalises this composition. It specifies the five stages of the structural layer, the stateful controller and its working memory, and the evidence-pool interface through which the two layers communicate.

# CHAPTER 3

# METHODOLOGY

The framework introduced in Chapter 1 has two layers, and this chapter specifies both. The structural layer is a deterministic pipeline that prepares a structured corpus for search and that returns, for any single query, a small set of citation-grade passages. The stateful agentic layer sits on top and turns a single retrieval call into a sequence of retrieval calls coordinated by a controller that holds a working memory. The interface between the two layers is the central object of the chapter, because it is what makes structure and state composable without each layer having to know the internal details of the other. Specific parameter values are stated abstractly in this chapter; the concrete values used in the empirical evaluation appear in Chapter 4.

## 3.1 Framework Overview

The framework has two phases. An *offline* phase prepares the corpus for search. An *online* phase processes each query. The structural layer covers both phases. The stateful agentic layer sits inside the online phase and wraps a controller around the retrieve-rerank-generate sequence. Figure 3.1 shows the two phases together with the dashed feedback arc through which the stateful layer reopens the online sequence with a refined query.

The interface between the two layers is the *evidence pool*. The structural layer produces, for each call from the agent, a set of retrieved passages tagged with provenance metadata. The stateful layer consumes the evidence pool, decides whether to act on it (answer, refine the query, expand the search, or stop), and calls the structural layer again with an updated query if needed. The structural layer has no notion of state; the stateful layer has no notion of how retrieval is implemented. The two interact only through queries in and evidence out.

## 3.2 Structural Layer

The structural layer is a five-stage pipeline: (*i*) document processing, (*ii*) metadata extraction, (*iii*) hybrid retrieval, (*iv*) cross-encoder reranking with metadata provenance, and (*v*) schema-constrained generation. The five stages share a single data type, the *chunk*, defined below. The notation used throughout this chapter is collected in Table 3.1.

### 3.2.1 Document Processing and Structure-Aware Chunking

A chunk is a contiguous span of text from a single source document, together with the metadata that identifies its position in the document hierarchy. The hierarchy is corpus-dependent. For a statutory corpus it is the chain (act, section, subsection, clause). For a clinical guideline it is the chain (guideline, chapter, recommendation). For a technical standard it is the chain (standard, clause, subclause). In each case the structural layer assumes that the hierarchy is recoverable from the source text by surface cues, either headings in plain text or tagged elements in a structured format.

Fixed-size chunking splits a document at every $N$ tokens, with no regard for these cues. The result is a stream of chunks that crosses hierarchy boundaries. A chunk that begins inside one

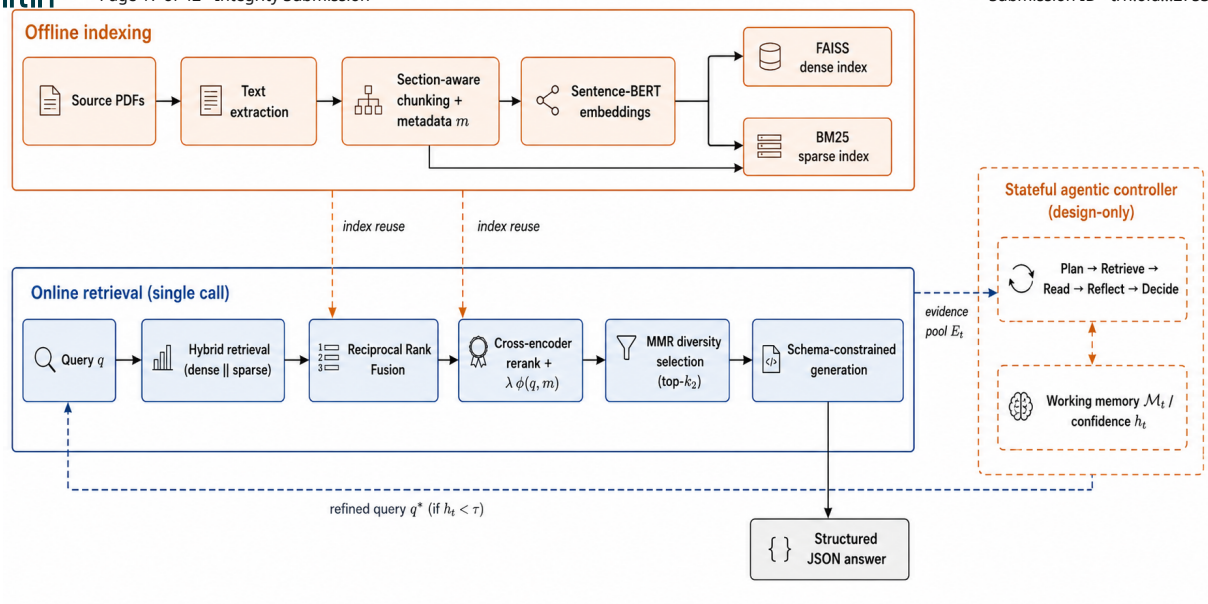


**Figure 3.1.** Offline indexing and online retrieval pipeline.

section and ends inside the next cannot be used as a citation, even if its embedding is close to the query, because the citation metadata is ambiguous. The structural layer replaces fixed-size chunking with a *priority-ordered separator* hierarchy. The splitter starts at the highest-priority boundary and only falls through to a lower one when the current chunk is still longer than the target size. Table 3.2 lists the priorities used for statutory text; the same priority list generalises to other hierarchical corpora by substituting the boundary tokens.

The target token budget is set by the encoder's maximum input length, with a small fixed overlap between adjacent chunks to preserve sentences that span a split. The overlap is bounded so that no chunk crosses a citation boundary, even though adjacent chunks share a small number of tokens around the split point. Each chunk therefore remains a self-contained citation unit, and the overlap acts purely as a sentence-preserving margin. Concrete values for the budget and the overlap appear in Chapter 4. Figure 3.2 contrasts the two cutting strategies on a short example document.

### 3.2.2 Metadata Extraction

Each chunk is tagged with metadata extracted by pattern matching on the source text. The patterns are corpus-specific. For statutory text the patterns capture:

- the act name, taken from the document filename and confirmed by in-text references;
- the section number, captured by regular expressions of the form `Section\s+\d+[A-Z]*`;
- the depth of the chunk in the hierarchy, computed from the prefix sequence of headings;
- the file-level location (filename, page, position within page) so that any citation produced downstream can be traced back to its origin.

The metadata is stored alongside the chunk text and the chunk embedding. It is consumed by two later stages. The reranker uses metadata as a tiebreaker, described in Section 3.2.5. The generator uses metadata to label each retrieved passage in the prompt so that any citation it emits can be checked against the source.

**Table 3.1.** Symbols used in this chapter.

| Symbol | Meaning |
|---|---|
| $q$ | input query |
| $c$ | a chunk (a passage of source text) |
| $m$ | metadata attached to a chunk |
| $e$ | dense embedding vector of a query or chunk |
| $N$ | total number of chunks in the corpus |
| $f(t,c)$ | frequency of term $t$ in chunk $c$ |
| $n_t$ | number of chunks containing term $t$ |
| $\lvert c\rvert$, $\bar{c}$ | chunk length and mean chunk length over the corpus |
| $k_1$, $b$ | saturation and length-normalisation parameters of BM25 (Eq. 3.2) |
| $\mathscr{R}_{\text{dense}}$, $\mathscr{R}_{\text{sparse}}$ | ranked lists returned by the dense and sparse retrievers |
| $\text{rank}_r(c)$ | rank of chunk $c$ in ranked list $r$ |
| $k_{\text{ret}}$ | top-$k$ depth used by each retriever |
| $k$ (in RRF) | smoothing constant of Eq. 3.3 |
| $\mathscr{C}_{\text{fused}}$ | candidate pool after Reciprocal Rank Fusion |
| $s_{\text{rerank}}$, $s_{\text{final}}$ | cross-encoder score and combined rerank+provenance score |
| $\varphi(q,m)$ | additive provenance score over metadata fields |
| $\lambda$ | provenance weight in Eq. 3.4 |
| $\text{sim}(c_i,c_j)$ | cosine similarity between chunk embeddings |
| $S$ | set of chunks already selected by MMR |
| $\lambda_{\text{mmr}}$ | relevance / diversity balance in Eq. 3.5 |
| $k_2$ | number of final contexts handed to the generator |
| $\tau$ | confidence threshold for the agentic loop |
| $T_{\max}$ | iteration budget for the agentic loop |
| $\mathscr{M}_t$ | working memory at iteration $t$ |
| $\mathscr{Q}_t$ | sequence of refined queries at iteration $t$ |
| $\mathscr{E}_t$ | evidence pool at iteration $t$ |
| $\mathscr{G}_t$ | set of unresolved sub-questions at iteration $t$ |
| $h_t$ | confidence estimate at iteration $t$ |
| $w_{\text{cov}}$, $w_{\text{agr}}$ | weights on coverage and agreement in $h_t$ (Section 3.3.3) |

### 3.2.3 Embedding and Indexing

The structural layer maintains two indexes over the chunks. A dense index supports semantic search; a sparse index supports exact-term search.

The dense index uses a sentence encoder [35] that maps each chunk to a fixed-length vector and applies the L2 normalisation in Equation 3.1, so that inner product equals cosine similarity:

$$e_{\text{norm}} = \frac{e}{\|e\|_2}. \tag{3.1}$$

The normalised vectors are stored in a FAISS inner-product index [42]. For corpora up to $10^5$ vectors the exact flat index has acceptable latency, and the framework uses it. For larger corpora the same dense retriever can be re-indexed with HNSW [43].

**Table 3.2.** Separator priority list for structure-aware chunking.

| Priority | Separator | Role |
|---|---|---|
| 1 | `\n\nSection` | primary statutory division |
| 2 | `\n\nArticle` | constitutional provision |
| 3 | `\n\nClause` | sub-provision |
| 4 | `\n\nRule` | procedural rule |
| 5 | `\n\nSchedule` | statutory appendix |
| 6 | paragraph break | fallback |
| 7 | sentence boundary | fallback |

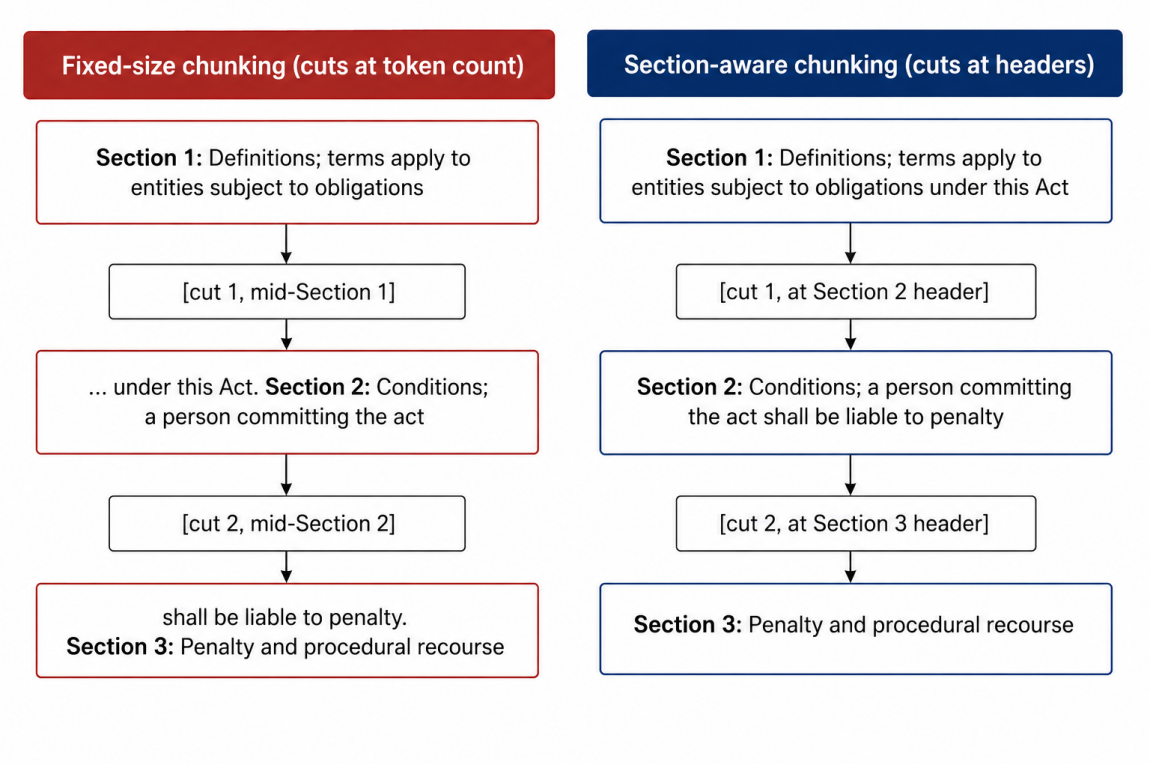


**Figure 3.2.** Fixed-size chunking versus section-aware chunking.

The sparse index uses Okapi BM25 [10] over the same chunk texts:

$$\mathrm{BM25}(q,c) = \sum_{t \in q} \mathrm{IDF}(t) \cdot \frac{f(t,c)\,(k_1+1)}{f(t,c) + k_1\left(1 - b + b\,\frac{|c|}{\bar{c}}\right)}, \tag{3.2}$$

where $f(t,c)$ is the frequency of term $t$ in chunk $c$, $|c|$ is the token length of $c$, $\bar{c}$ is the mean chunk length over the corpus, $k_1$ and $b$ are the term-frequency saturation and length-normalisation parameters of [10], and the inverse document frequency (IDF) of term $t$ is the non-negative variant

$$\mathrm{IDF}(t) = \ln\left(1 + \frac{N - n_t + 0.5}{n_t + 0.5}\right),$$

with $N$ the chunk count and $n_t$ the number of chunks containing $t$. The inner "+1" shifts the canonical Okapi IDF into the non-negative range and matches the BM25 implementation used in widely deployed full-text search engines, including the implementation that is used by the sparse index in this thesis. BM25 is run at chunk granularity, so $N$ and $n_t$ are chunk-level statistics rather than document-level statistics.

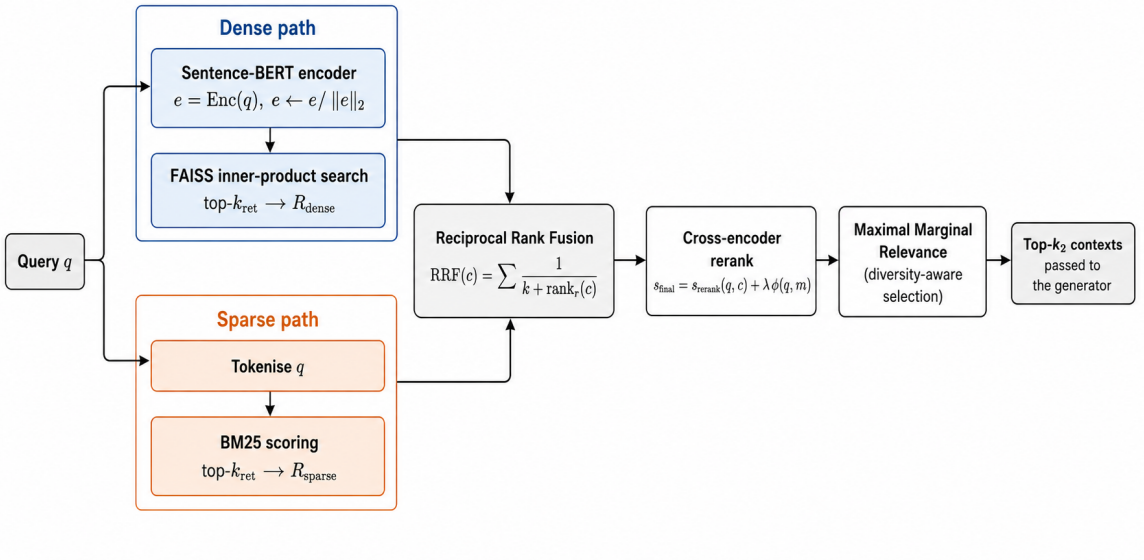


**Figure 3.3.** Hybrid retrieval cascade with rank fusion and reranking.

### 3.2.4 Hybrid Retrieval with Rank Fusion

Each retrieval call runs the dense index and the sparse index in parallel. Both return their own top-$k_{\mathrm{ret}}$ ranked list. The two lists are merged through Reciprocal Rank Fusion [13], which combines ranks rather than scores and therefore avoids the distribution-mismatch problem between inner products and BM25 scores:

$$\mathrm{RRF}(c) \;=\; \sum_{r \in \{\mathscr{R}_{\mathrm{dense}}, \mathscr{R}_{\mathrm{sparse}}\}} \frac{1}{k + \mathrm{rank}_r(c)}, \tag{3.3}$$

where $\mathrm{rank}_r(c) = \infty$ when chunk $c$ does not appear in ranked list $r$ (so the missing list contributes zero), and $k$ is a small smoothing constant [13]. The top-$k_{\mathrm{ret}}$ chunks by RRF score form the candidate pool $\mathscr{C}_{\mathrm{fused}}$ that is handed to the reranker. The pool size is capped at the retrieval depth even when the two source lists overlap heavily, so that the latency of the next stage stays bounded. Figure 3.3 summarises the cascade from the query to the top-$k_2$ contexts passed to the generator.

### 3.2.5 Cross-Encoder Reranking with Metadata Provenance

A cross-encoder [41] re-scores every candidate $c_i \in \mathscr{C}_{\mathrm{fused}}$ by feeding the query and the candidate as a single input. Unlike a bi-encoder, a cross-encoder lets every token of the query attend to every token of the candidate, which improves precision at the cost of an $O(|\mathscr{C}_{\mathrm{fused}}|)$ runtime.

The cross-encoder is web-trained and has not seen the specialist corpus. To compensate, the framework adds a metadata provenance score that the reranker would otherwise be unable to exploit:

$$s_{\mathrm{final}}(q, c_i) \;=\; s_{\mathrm{rerank}}(q, c_i) \;+\; \lambda\, \varphi(q, m_i), \tag{3.4}$$

where $m_i$ is the metadata attached to chunk $c_i$, and $\varphi(q, m_i)$ is an additive provenance score over matchable metadata fields. For a statutory corpus, two fields matter:

- a section-number match (the query mentions a section number that also appears in the chunk's metadata) contributes a high weight, because section numbers are unambiguous identifiers;
- an act-name match (the query mentions an act whose name is recoverable from the chunk's metadata) contributes a lower weight.

**Algorithm 1** Structural retrieval cascade.

**Require:** query $q$, dense index $\mathscr{I}_d$, sparse index $\mathscr{I}_s$, parameters $k_{\text{ret}}$, $k_2$, $k$, $\lambda$, $\lambda_{\text{mmr}}$

$e \leftarrow \textsc{Embed}(q)$; $e \leftarrow e/\|e\|_2$
$\mathscr{C}_d \leftarrow \textsc{TopK}(\mathscr{I}_d, e, k_{\text{ret}})$
$\mathscr{C}_s \leftarrow \textsc{TopK}(\mathscr{I}_s, \text{BM25}(q,\cdot), k_{\text{ret}})$
$\mathscr{C}_{\text{fused}} \leftarrow \textsc{TopByRRF}(\mathscr{C}_d, \mathscr{C}_s, k)[1..k_{\text{ret}}]$
**for all** $c \in \mathscr{C}_{\text{fused}}$ **do**
    $s_{\text{rerank}}(c) \leftarrow \textsc{CrossEncoder}(q, c)$
    $s_{\text{final}}(c) \leftarrow s_{\text{rerank}}(c) + \lambda \cdot \varphi(q, m(c))$
**end for**
$\mathscr{C}_{\text{fused}} \leftarrow$ *sort by* $s_{\text{final}}$ *descending*
$S \leftarrow \emptyset$
**while** $|S| < k_2$ **do**
    $c^* \leftarrow \arg\max_{c \in \mathscr{C}_{\text{fused}} \setminus S} \left[\lambda_{\text{mmr}}\, s_{\text{final}}(c) - (1-\lambda_{\text{mmr}}) \max_{c' \in S} \text{sim}(c, c')\right]$
    $S \leftarrow S \cup \{c^*\}$
**end while**
**return** $S$

The weight $\lambda$ is chosen on a development split so that the maximum provenance bonus is small relative to the spread of the cross-encoder's own scores. The bonus then breaks ties between candidates that the cross-encoder considers nearly equivalent; it is not a dominant signal.

### 3.2.6 Diversity-Aware Selection

Candidates ranked by $s_{\text{final}}$ are passed to a Maximal Marginal Relevance (MMR) selector [14]. Without a diversity step, the top-$k_2$ candidates can all come from the same section of the corpus, even when the query references several. MMR maximises a convex combination of relevance to the query and dissimilarity to already-selected chunks:

$$c^* = \underset{c_i \in \mathscr{C}_{\text{fused}} \setminus S}{\arg\max} \left[\lambda_{\text{mmr}}\, s_{\text{final}}(q, c_i) - (1-\lambda_{\text{mmr}}) \max_{c_j \in S} \text{sim}(c_i, c_j)\right], \tag{3.5}$$

where $S$ is the running set of selected chunks, sim is cosine similarity in the dense embedding space, and $\lambda_{\text{mmr}} \in [0,1]$ trades off relevance against diversity. The final top-$k_2$ chunks selected by this rule become the context that is passed to the generator.

The full structural cascade is summarised in Algorithm 1. It takes a query and returns a diverse, reranked set of $k_2$ chunks together with their metadata; the generation step that consumes the chunks is treated separately in Section 3.2.7.

### 3.2.7 Schema-Constrained Generation

The generator is an off-the-shelf instruction-tuned language model. The prompt contains three parts: a fixed instruction that forbids citations to sources outside the retrieved context, the retrieved chunks with their provenance metadata, and the output schema. The schema is a JavaScript Object Notation (JSON) object whose shape is fixed by the application. In abstract form it declares two role groups and a free-text note: a fixed set of *entity fields* that identify the parties named in the query, an array of *citation records* that ground the answer, each with a source identifier, a reference within that source, a free text description, and a set of application-specific attribute fields that follow the citation grain of the corpus:

```
{ "<entity_role_1>": "string",
```

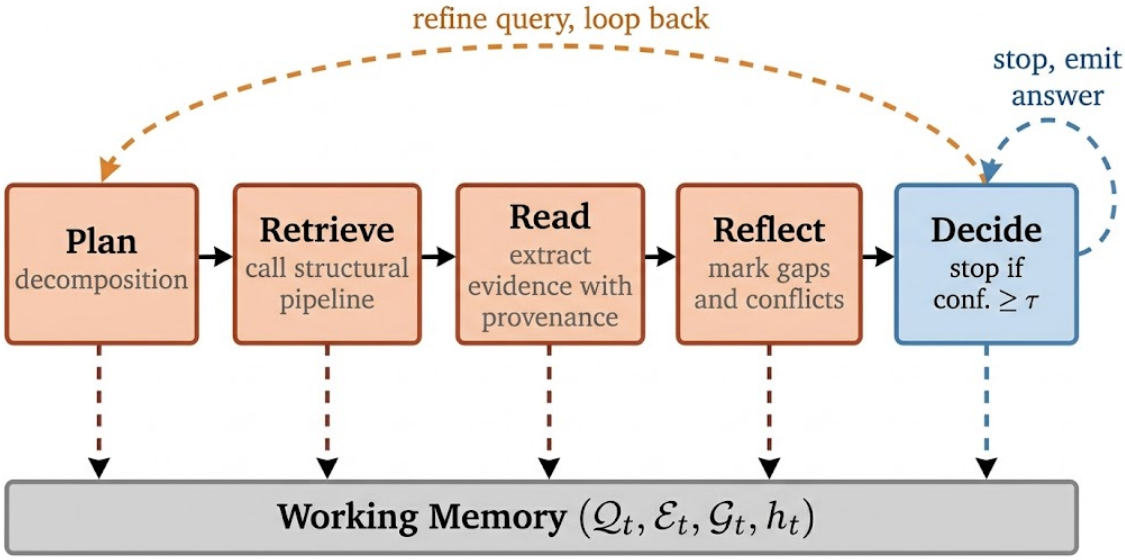


**Figure 3.4.** Stateful agentic control loop

```
"<entity_role_2>": "string",
"citations": [
  { "source": "string",
    "reference": "string",
    "description": "string",
    "<attr_1>": "string",
    "<attr_2>": "string"
  }, ...
],
"note": "string" }
```

For any concrete instantiation, the entity-role keys and the per-citation attribute keys are fixed by the application schema; the `citations` array name itself may be renamed without loss of generality. The instantiation used in the empirical evaluation appears in Chapter 4.

The response is parsed by a four-pass parser that tries, in order: direct JSON parsing of the entire response, extraction of the JSON block delimited by code fences, brace-matching from the first to the last balanced brace, and removal of duplicate keys inside any object. If all four passes fail, the system retries with an exponentially increasing backoff. The combination of schema constraints and the parser eliminates a class of failures in which the model returns fluent prose that is not parseable into the required fields.

## 3.3 Stateful Agentic Layer

The structural layer issues a single retrieval call. The stateful layer wraps a controller around the structural layer so that retrieval becomes one action inside a longer reasoning loop. The controller draws on the agentic methods reviewed in Section 2.3: action and observation loops [7], self-reflection over intermediate output [11, 12], retrieval-side self-reflection [6], and confidence-triggered re-retrieval [5]. This section specifies the composition of these methods, the interface between the controller and the structural layer, and the stopping rule. The five operations and the working memory bar are summarised in Figure 3.4.

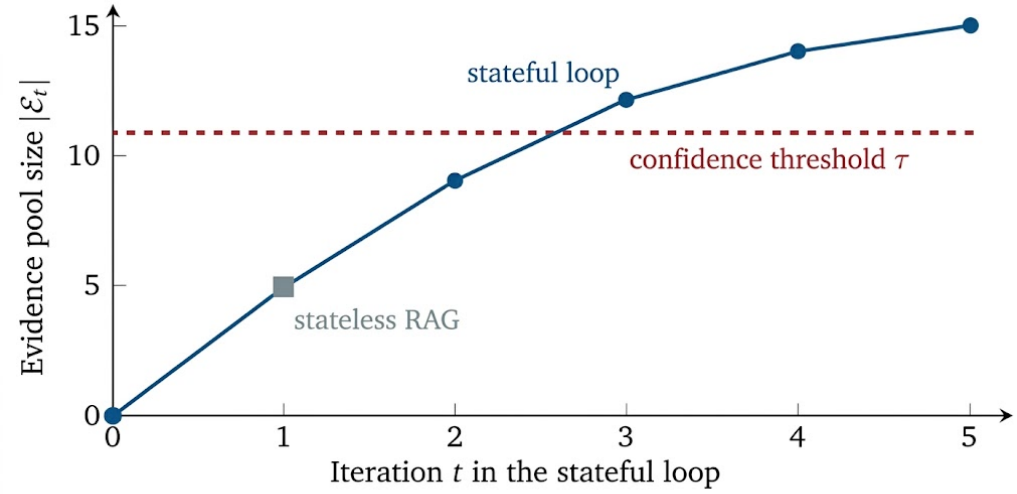


**Figure 3.5.** Schematic evidence-pool growth across iterations of the stateful loop.

### 3.3.1 State Representation

The working memory $\mathcal{M}_t$ at iteration $t$ is a tuple $(\mathcal{Q}_t, \mathcal{E}_t, \mathcal{G}_t, h_t)$ that contains the elements listed below. Figure 3.5 illustrates schematically how the evidence pool component $\mathcal{E}_t$ accumulates as iterations proceed and how the stopping rule cuts the loop off once the confidence estimate crosses the threshold $\tau$; the curve is intended to depict the controller's design behaviour and is not derived from empirical runs.

- $\mathcal{Q}_t$, the original query plus the sequence of refined queries issued so far;
- $\mathcal{E}_t$, the evidence pool, a set of chunks (with provenance metadata) accumulated across calls to the structural layer;
- $\mathcal{G}_t$, the set of *unresolved sub-questions* at iteration $t$. A sub-question is unresolved if it has either no supporting proposition in the current evidence pool (the *open* case) or two propositions that contradict each other (the *contradicted* case);
- $h_t$, a confidence estimate for the current best draft answer.

The working memory is updated by each operation in the loop. Nothing in the loop is discarded between iterations, so an answer that depends on evidence retrieved in iteration 1 but only becomes relevant in iteration 3 is still available to the generator.

### 3.3.2 Action Set

The controller has five operations, composed for this framework from the agentic literature reviewed in Section 2.3:

1. **Plan.** Read the working memory, list the sub-questions implied by the original query, and pick the next sub-question to address.
2. **Retrieve.** Call the structural pipeline with the chosen sub-question. The pipeline returns a top-$k_2$ list of chunks with metadata. The chunks are added to the evidence pool $\mathcal{E}_{t+1}$.
3. **Read.** For each newly retrieved chunk, extract the propositions relevant to the sub-question and bind them to the chunk's provenance metadata.
4. **Reflect.** Compare the current evidence pool against the original sub-question list. Mark a sub-question as *covered* if at least one proposition supports it, *contradicted* if two propo-

sitions disagree, and *open* otherwise. Set $\mathcal{G}_{t+1}$ to the union of the open and contradicted sub-questions; covered sub-questions drop out of the unresolved set.

5. **Decide.** Compute a confidence estimate $h_{t+1}$ from the current evidence pool (defined formally below in Section 3.3.3). The controller continues to the next iteration when $h_{t+1} < \tau$, the iteration budget has not been reached, and at least one sub-question remains in $\mathcal{G}_{t+1}$. It terminates and hands control to the final generation step when any of three stopping conditions hold: (a) $h_{t+1} \geq \tau$ (sufficient confidence); (b) $\mathcal{G}_{t+1} = \emptyset$ (every sub-question is covered and none is contradicted); or (c) the iteration counter reaches the budget, $t+1 \geq T_{\max}$. In case (c) the residual unresolved sub-questions are passed forward to the generator and reported alongside the structured answer so that any remaining uncertainty is explicit.

The Plan step corresponds to the planning step in ReAct-style agents [7]. The Read and Reflect steps implement the propose-and-criticise pattern of self-refinement [11, 12]. The Decide step combines the confidence-triggered re-retrieval idea of FLARE [5] with the self-gating idea of Self-RAG [6]. The contribution of this section is therefore not any single step but the composition of all five over a working memory that the structural layer's metadata-rich chunks make tractable.

### 3.3.3 Confidence Estimation and the Stopping Rule

The confidence estimate $h_t$ is a non-negative weighted combination of two signals computed from the working memory $\mathcal{M}_t$ alone, without invoking the generator inside the loop:

- *Coverage*, the fraction of sub-questions in the original $\mathcal{G}_0$ that have at least one supporting proposition in the current pool;
- *Agreement*, the fraction of those covered sub-questions whose supporting propositions do not contradict each other.

The two are combined linearly with non-negative weights $w_{\text{cov}}$ and $w_{\text{agr}}$ that satisfy $w_{\text{cov}} + w_{\text{agr}} = 1$, so that $h_t \in [0,1]$. The weights are tuned on a held-out development split to maximise agreement between the controller's stopping decision and a reference set of sufficiency labels; the calibration procedure is specified at the same level of abstraction as the rest of the agentic layer and the empirical fit is left to subsequent work (Section 6.3).

The loop terminates when the updated estimate $h_{t+1}$ reaches the threshold $\tau$, when no unresolved sub-questions remain, or when the iteration count reaches the budget $T_{\max}$. The budget bounds latency in the worst case; the threshold lets low-complexity queries return in one iteration. Monotonicity of $h_t$ is not assumed: if reflection in iteration $t+1$ uncovers a contradiction, $h_{t+1}$ may fall below $h_t$. Termination is therefore guaranteed by $T_{\max}$ and not by the confidence sequence.

### 3.3.4 Algorithmic Form

Algorithm 2 states the loop in algorithmic form for reference. The structural-layer call inside the loop is the five-stage pipeline of Section 3.2; the loop is otherwise self-contained.

## 3.4 Composition of the Structural and Stateful Layers

The structural layer determines which passages the generator receives for a single query. The stateful layer determines the next query, given the evidence collected so far. The two layers communicate only through the evidence pool and the query stream, and this separation makes either layer replaceable without touching the other. A different retriever (with a different encoder, fusion rule, or reranker) can be substituted in the structural layer with no change to the

**Algorithm 2** Stateful agentic control loop (algorithmic form).

**Require:** query $q_0$, structural retriever RETRIEVE, generator GEN, threshold $\tau$, iteration budget $T_{\max}$

$\mathcal{Q}_0 \leftarrow \{q_0\}$; $\mathcal{E}_0 \leftarrow \emptyset$; $\mathcal{G}_0 \leftarrow \textsc{Plan}(q_0)$; $h_0 \leftarrow 0$; $t \leftarrow 0$
**while** $h_t < \tau$ **and** $\mathcal{G}_t \neq \emptyset$ **and** $t < T_{\max}$ **do**
  $q^* \leftarrow \textsc{ChooseGap}(\mathcal{G}_t)$
  $C \leftarrow \textsc{Retrieve}(q^*)$ ▷ calls the structural layer
  $\mathcal{E}_{t+1} \leftarrow \mathcal{E}_t \cup C$
  $P \leftarrow \textsc{Read}(C, q^*)$ ▷ propositions with provenance
  $\mathcal{G}_{t+1} \leftarrow \textsc{Reflect}(\mathcal{G}_t, P)$
  $\mathcal{Q}_{t+1} \leftarrow \mathcal{Q}_t \cup \{q^*\}$
  $h_{t+1} \leftarrow \textsc{Confidence}(\mathcal{G}_{t+1}, \mathcal{E}_{t+1})$
  $t \leftarrow t+1$
**end while**
$y \leftarrow \textsc{Gen}(q_0, \mathcal{E}_t, \mathcal{G}_t)$ ▷ single generation at termination
**return** $y$

agentic controller; a different controller (without reflection, or with parallel queries) can be substituted in the stateful layer with no change to the structural layer.

Chapter 4 instantiates the abstract symbols introduced above. It fixes the corpus, the off-the-shelf neural components, the hyperparameters of the structural layer, the concrete output schema, the expert-annotated benchmark, and the metrics used in the empirical evaluation.

# CHAPTER 4

# EXPERIMENTAL SETUP

The empirical evaluation in the thesis is designed to test the structural layer of the framework on a corpus that exhibits the four properties of a knowledge-intensive corpus laid out in Section 1.1.1: large size, hierarchical structure, periodic revision, and strict citation grounding. Indian statutory text satisfies all four properties and is the concrete instantiation used here. All neural components are off-the-shelf, so the reported numbers reflect the framework's behaviour in the absence of any domain-specific fine-tuning of the encoder, reranker, or generator.

## 4.1 Evaluation Strategy

The framework has two layers. The structural layer is fully implemented and is the subject of the empirical evaluation in this chapter and in Chapter 5. The stateful agentic layer is a design, presented in Section 3.3, and its end-to-end evaluation against the same benchmark is left to subsequent work (Section 6.3). Every number reported in this thesis is therefore a number produced by the structural layer operating in a single retrieval call per query.

Two practical constraints shape the evaluation. First, no public benchmark exists that scores retrieval-augmented systems on the specific task of citing the right provision in a structured authority corpus. The thesis therefore constructs one, with ground truth produced independently by a domain expert. Second, the structural layer has several parameters (chunk size, overlap, retrieval depth, fusion constant, provenance weight, MMR balance, diversity cut-off) whose interaction matters. Parameters are chosen on a held-out development split, and the test set is held out until the evaluation reported in Chapter 5.

## 4.2 Corpus

The corpus used in the evaluation has to exhibit the four properties of a knowledge-intensive corpus (Section 1.1.1): large enough to defeat any language-model context window, hierarchically structured, periodically revised, and citation-grounded. Indian central legislation satisfies all four properties and is used as one concrete instantiation of the framework.

### 4.2.1 Collection

The corpus consists of forty central-level Indian legal documents sourced from official government repositories maintained by legislative and regulatory bodies. The collection spans criminal law, civil law, constitutional provisions, taxation, intellectual property, labour, environmental law, and several specialised regulatory acts. The collection includes both the statutes that came into force in mid-2024 (the Bharatiya Nyaya Sanhita, 2023; the Bharatiya Nagarik Suraksha Sanhita, 2023; the Bharatiya Sakshya Adhiniyam, 2023) and the older statutes that remain operative. Per-category document counts are not separated in the source and are therefore not reported here.

**Table 4.1.** Corpus statistics after structure-aware chunking.

| Quantity | Value |
|---|---|
| Documents indexed | 40 |
| Total PDF pages | ≈ 3,500 |
| Pages that failed extraction | 23 (0.66%) |
| Chunks produced | 20,836 |
| Mean chunk length (tokens) | 647 |
| Standard deviation of chunk length | 118 |
| Embedding dimension | 768 |
| FAISS index size on disk | 235 MB |
| Chunks tagged with act name | 20,565 (98.7%) |
| Chunks tagged with section number | 12,937 (62.1%) |

### 4.2.2 Preprocessing and Chunk Statistics

Text extraction uses a parsing pipeline based on a Portable Document Format (PDF) library, with four worker processes. PDF parsing of the full collection of approximately 3,500 pages completes in roughly twelve minutes wall-clock on commodity hardware (an eight-core consumer-grade CPU, no GPU). Index construction on top of the extracted text takes a comparable amount of time, reported separately in Section 4.3.3. Twenty-three pages out of approximately 3,500 (0.66%) fail to extract because the source PDF contains the page as a scanned image with no embedded text. These pages are excluded from the index; the small fraction of pages affected does not materially change the coverage of the corpus.

Structure-aware chunking, configured as in Section 3.2.1, produces 20,836 chunks. Table 4.1 summarises the resulting collection statistics.

The mean chunk length is 7.6% below the 700-token target. The shortfall comes from two sources. Most chunks end at a section boundary that arrives before the budget is reached, which is the intended behaviour of the priority-ordered splitter. A smaller group of chunks ends at a fallback boundary because the section itself is longer than the budget and is split at a lower-priority separator. Section 5.5 discusses the second group.

The act-name extraction rate of 98.7% indicates that the filename and the in-text references are sufficient for almost every chunk. The section-number extraction rate of 62.1% is lower because preambles, definitions sections, and schedules do not follow the `Section N` naming convention. The 37.9% of chunks without a section number are still retrievable through the dense and sparse paths but cannot benefit from the section-number provenance bonus described in Section 3.2.5.

## 4.3 Implementation

The framework of Chapter 3 reduces to a set of software components. This section names the off-the-shelf neural components used, fixes the values of the hyperparameters introduced in the abstract framework, and notes the hardware and software stack on which the experiments were run. None of the neural components is fine-tuned on the corpus; the structural layer is therefore the sole source of any domain-specific behaviour.

**Table 4.2.** Hyperparameter settings used in the evaluation.

| Symbol | Role | Value |
|---|---|---|
| Chunk size | target tokens per chunk | 700 |
| Chunk overlap | tokens of overlap (12.1%) | 85 |
| Embedding dim. | dense vector dimensionality | 768 |
| $k_{\text{ret}}$ | top-$k$ from each retriever | 20 |
| Fusion $k$ (RRF) | smoothing constant in Eq. 3.3 | 60 |
| $\lambda$ | provenance weight in Eq. 3.4 | 0.15 |
| $\lambda_{\text{mmr}}$ | balance in Eq. 3.5 | 0.5 |
| $k_2$ | final contexts for generation | 5 |
| Provenance: section match | weight added by section-number agreement | 1.0 |
| Provenance: act match | weight added by act-name agreement | 0.5 |
| Generator temperature | sampling temperature | 0.0 |
| Generator max tokens | output budget | 8,192 |
| Context truncation | characters per retrieved context | 2,000 |

### 4.3.1 Models

The structural layer uses three off-the-shelf neural components. None of them is fine-tuned on the corpus.

- **Sentence encoder.** A general-purpose Sentence-BERT encoder [35], released as the public checkpoint `sentence-transformers/all-mpnet-base-v2`, produces 768-dimensional embeddings. No public encoder fine-tuned on Indian statutory text was available at the time the experiments were conducted.
- **Cross-encoder reranker.** A cross-encoder trained on the Microsoft Machine Reading Comprehension (MS MARCO) passage-ranking dataset [41] re-scores the fused candidates. The implementation uses the public `cross-encoder/ms-marco-MiniLM-L6-v2` checkpoint. Its size keeps the reranking latency well below the dense retrieval latency at the candidate counts used here.
- **Generator.** An instruction-tuned decoder language model accessed through an OpenAI-compatible application programming interface (API) endpoint. Generation runs at temperature 0 for reproducibility, with an output token limit of 8,192.

### 4.3.2 Hyperparameters

Table 4.2 lists the hyperparameters used in the empirical evaluation. The values for $\lambda$ (provenance weight), $k_{\text{ret}}$ (retrieval depth), and the chunk size were selected on a development split that is disjoint from the test complaints described in Section 4.4.

The provenance weight $\lambda = 0.15$ keeps the maximum provenance bonus at $\lambda \cdot 1.5 = 0.225$, which is small relative to the typical spread of cross-encoder logits over the candidate pool. The bonus therefore acts as a tiebreaker in the sense of Section 3.2.5 rather than as a dominant signal.

### 4.3.3 Hardware and Software

The dense index is built with FAISS [42] in the exact `IndexFlatIP` configuration. The sparse index is built with a standard implementation of Okapi BM25 [10]. The pipeline runs on a commodity workstation with an eight-core CPU; no GPU is required at query time because

the cross-encoder is small. Index construction on top of the extracted text takes approximately twelve minutes wall-clock on this hardware, separate from the PDF-parsing time reported in Section 4.2.2.

### 4.3.4 Output Schema for the Statutory Instantiation

The abstract output schema described in Section 3.2.7 is instantiated for the statutory corpus by fixing the entity-role keys to `victim` and `culprit`, renaming the `citations` array to `charges`, and populating the per-citation attribute keys with the procedural attributes a statutory citation carries. The concrete schema used in this evaluation is

```
{ "victim": "string",
  "culprit": "string",
  "charges": [
    { "law": "string",
      "description": "string",
      "bailable": "Yes/No",
      "cognizable": "Yes/No" }
  ],
  "note": "string" }
```

The two entity-role keys (`victim`, `culprit`) identify the parties named in the complaint. Each charge object plays the role of a citation record: `law` and the `description` string supply the source identifier and the free-text summary mandated by the abstract schema; the `bailable` and `cognizable` keys are the application-specific procedural attributes contributed by the statutory domain. The root-level `note` field carries any residual free-text commentary that does not fit into a per-charge record.

## 4.4 Expert-Annotated Benchmark

The benchmark consists of two hundred legal complaints in natural language. The complaints were collected from legal aid centres and public forums, and span the statutory domains represented in the corpus, including criminal, civil, and environmental categories. Each complaint is a factual account of a dispute. Some include explicit statutory references; others are expressed in informal language with no legal terminology.

Each complaint was reviewed by a single practising lawyer with at least two years of experience in Indian law. For each complaint the lawyer produced a reference analysis that lists the applicable provisions and identifies the *primary* applicable section: the one provision most directly invoked by the facts. The reference analysis is the ground truth against which model outputs are scored. The annotation protocol fixed the schema of the reference analysis so that comparison with the model's structured output is unambiguous.

## 4.5 Evaluation Metrics

Two criteria are scored against the lawyer's reference analysis.

**Primary citation identification.** A prediction is correct if the model's top-ranked section, as ranked in the `charges` array of the structured output, exactly matches the section that the lawyer marked as primary. Accuracy is the fraction of complaints with a correct prediction.

**Complete citation listing.** A prediction is correct if the set of sections returned by the model is exactly the set of sections in the lawyer's reference analysis. The set is unordered, but it must

be exact: a missing section, an extra section, or a section attributed to the wrong act counts as an error. Accuracy is the fraction of complaints with a correct set.

For both criteria the thesis reports the point estimate and a 95% Wilson score confidence interval [50]. The Wilson interval is appropriate here because the observed proportions are close to the boundaries of the unit interval, where the standard normal approximation to the binomial is inaccurate at moderate sample sizes. The Wilson interval is asymmetric around the point estimate, which reflects the asymmetry of the binomial distribution near those boundaries.

Latency is not reported as a primary metric because the system runs in a single retrieval call per query and the dominant cost is the cross-encoder, whose runtime is fixed once the candidate pool is fixed. A non-instrumented timing check on a sample of queries placed end-to-end latency on the order of one to several seconds per query on the hardware described above; this is reported as a qualitative observation rather than a benchmarked measurement.

# CHAPTER 5

# RESULTS AND DISCUSSION

The empirical evaluation bears on two questions that the framework of Chapter 3 raises about contextual reasoning over a knowledge-intensive corpus. First, how much of the default pipeline's failure on a structured corpus can be removed by respecting document hierarchy at every stage of a single retrieval call. Second, what residual error remains after the structural layer has done its work, and whether that residual error has the shape that the stateful layer is designed to address. The empirical evidence answers the first question in the positive and characterises the second.

## 5.1 Component-Level Observations

The claim that the structural pipeline preserves citation grain within a single retrieval call is testable one stage at a time. Chunking, hybrid retrieval, and generation are examined separately so that the residual error of the full system can be attributed to its source.

### 5.1.1 Chunking

Structure-aware chunking produced 20,836 chunks across the forty indexed documents, with a mean chunk length of 647 tokens and a standard deviation of 118 tokens. The mean falls 7.6% below the target of 700 tokens. Inspection of a sample of chunks confirmed that the splitter ended most chunks at a section or article boundary rather than mid-sentence. The exceptions are chunks belonging to procedural codes whose individual sections exceed the budget; these chunks fall through to the paragraph separator and the sentence separator in the priority list. The chunks affected by this fallback contribute a small share of the residual errors observed at the end-to-end level.

The metadata layer tagged 20,565 of the 20,836 chunks (98.7%) with an act name. The 1.3% that escaped tagging are chunks from front matter (cover pages, contents pages, schedules without explicit headings). Section-number tagging reached 12,937 chunks (62.1%). The remaining 37.9% are chunks from preambles, definitions sections, schedules, and unnumbered sections, which do not match the section-number pattern. This gap is, among the component-level limitations exposed by the evaluation, the one that accounts for the largest share of end-to-end errors; Section 5.3 traces a subset of those errors back to it.

### 5.1.2 Hybrid Retrieval

The two retrievers behave differently on the same queries, in a way that motivates the rank-fusion design.

- The dense retriever returns chunks that match the query by meaning. It recovers sections that use different wording from the complaint but address the same factual scenario. Its weakness, in line with the observations on specialist corpora reported in [8, 9], is on queries that contain a specific section number or act name, where surface form carries more signal than semantics.
- The sparse retriever returns chunks that share high-IDF terms with the query. It recovers

**Table 5.1.** End-to-end accuracy with 95% Wilson confidence intervals.

| Metric | Correct | Accuracy | 95% CI |
|---|---|---|---|
| Primary citation identification | 188 / 200 | 94.0% | [89.8%, 96.5%] |
| Complete citation listing | 164 / 200 | 82.0% | [76.1%, 86.7%] |

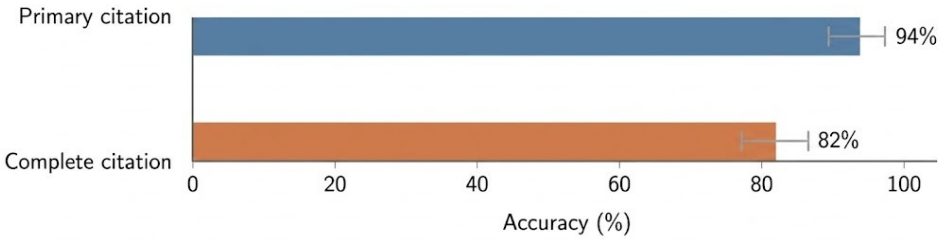


**Figure 5.1.** End-to-end accuracy on the expert-annotated benchmark.

sections when the complaint cites them by number or by act name, and it recovers chunks that contain rare legal phrases verbatim. Its weakness is on informal complaints with no statutory terminology, where the terms it ranks highly are content words that occur throughout the corpus.

The two failure cases are largely disjoint. Reciprocal Rank Fusion therefore aligns with this complementarity: it assigns a high fused score whenever a chunk ranks highly on either path and demotes only chunks that rank poorly on both. After fusion, the cross-encoder reranks the fused pool, and the metadata provenance bonus breaks ties in favour of candidates whose section number or act name appears in the query.

### 5.1.3 Generation

The generator's outputs conform to the required JSON schema on every evaluated query. The four-pass parser handles every output without invoking the backoff retry, which indicates that the schema constraint is enforced by the prompt at the level of the language model itself and that the parser is acting as a defence in depth rather than as the primary mechanism. A manual review of fifty randomly chosen outputs against the retrieved context found no citation to a provision absent from that context. This review is exploratory and is not a substitute for a full faithfulness evaluation against a held-out reference.

A second observation comes out of the periodic-revision property that the corpus was chosen to exhibit. The index contains both the provisions that came into force in mid-2024 and the older ones that they replaced. When the retrieved context contains a current provision, the generator cites the current provision. When the query refers to the older one by name, the retriever returns whichever version the index holds and the generator cites that. Neither behaviour requires any logic in the generator beyond the provenance metadata that the structural layer attaches to each chunk; this is the corpus revision case that the structural metadata is designed to handle.

## 5.2 End-to-End Accuracy of the Structural Pipeline

The structural layer, run as a single retrieval call per query, is evaluated on the full set of two hundred complaints. Table 5.1 reports the two scores and their 95% Wilson confidence intervals [50]; Figure 5.1 plots the same two scores.

The primary-citation score of 94.0% means that the citation the model puts first in its structured output matches the citation that the expert marked as primary in 188 of 200 cases. The complete-citation score of 82.0% means that the unordered set of citations in the model output

exactly matches the unordered set of citations in the expert's analysis in 164 of 200 cases. The two scores are connected on this benchmark: every test query that fails primary identification was also observed to fail complete listing, but the converse is not true.

## 5.3 Error Analysis

Twelve test queries fail primary identification. Thirty-six fail complete listing, and the twelve primary failures are a subset of the thirty-six listing failures. Manual inspection of the failing queries identifies three recurring failure modes.

1. **Cross-domain complaints.** A complaint that invokes provisions in more than one statutory domain (for example, an environmental incident with a criminal-intent component) is more difficult to retrieve in a single pass. The relevant provisions are scattered across acts that share little surface vocabulary, so the dense retriever has to bridge two encoder neighbourhoods at once. The sparse retriever helps when the complaint cites a section number, but not when the complaint is informal.

2. **Informal language.** A complaint expressed in everyday vocabulary, with no statutory terminology, reduces recall in both retrieval channels. The sparse retriever has no high-IDF terms to anchor on; the dense retriever has to bridge the vocabulary gap between informal description and formal statutory wording on its own.

3. **Missing section metadata.** A relevant chunk that lies in a preamble, a definitions section, or a schedule has no section-number metadata (see Section 5.1.1). The provenance bonus contributes zero to such a chunk's final score, and the candidate has to compete on the cross-encoder score alone. When the cross-encoder ranks it just below an alternative chunk, the answer omits it. This pattern appears in a substantial fraction of the observed listing errors.

The three failure modes are not mutually exclusive. A single failing query can exhibit all three at once, and the thesis does not break down the error counts by mode because such a breakdown would require an arbitration rule that the manual inspection does not provide.

## 5.4 Discussion

The twelve-point gap between primary accuracy (94.0%) and complete-citation accuracy (82.0%) is the empirical signature of single-pass retrieval on multi-source queries. Identifying a single primary citation can be supported by one strong top-ranked chunk; reproducing the full set of applicable citations requires the retriever to return every relevant chunk, the reranker to keep every relevant chunk above the top-$k_2$ cut-off, and the generator to emit every relevant chunk. A failure at any one stage produces a listing error without necessarily producing a primary error. The gap is therefore evidence for the core claim of this thesis: a one-shot retrieval call is too narrow a context window for a question that needs to compose evidence from several parts of the corpus.

The error analysis supports three implications for how a retrieval-augmented system should manage *context* and *memory* on a hierarchically structured corpus.

- **Context preservation comes from chunking, not from the retriever.** Once a chunk crosses a citation boundary, no retriever can recover the boundary. The structural layer's section-aware chunker is therefore the precondition for every later stage. The 98.7% act-tagging and 62.1% section-tagging figures are not metrics of the retrieval pipeline; they are metrics of how much of the corpus the chunker can hand to the pipeline as well-formed citation units.

- **The reranker's domain mismatch is a memory problem.** A web-trained cross-encoder does not encode the in-domain meaning of an act name or a section number. The additive

**Table 5.2.** Mapping of pipeline components to failure modes.

| Component | Targets | Failure mode mitigated | Status |
|---|---|---|---|
| Section-aware chunking | document hierarchy | structural blindness | evaluated |
| Metadata extraction | citation provenance | lost attribution after chunking | evaluated |
| Dense + sparse hybrid | encoder-corpus mismatch | single-channel recall gap | evaluated |
| Reciprocal Rank Fusion | heterogeneous score scales | unsafe score-level blending | evaluated |
| Provenance bonus | reranker domain bias | low score on metadata-rich chunks | evaluated |
| MMR diversity selection | top-$k$ redundancy | all results from same section | evaluated |
| Schema-constrained output | unsupported claims | hallucinated citations | evaluated |
| Stateful agentic loop | single-pass retrieval | cross-domain and informal queries | design only |

provenance bonus is a way of giving the reranker an *external memory* of the corpus's structure that its pre-training did not provide. The bonus offsets part of the cross-encoder's domain-transfer error but cannot promote chunks that have no extracted metadata at all.

- **The residual error matches the failure mode that a stateful layer is designed to address.** The two failure modes that survive the structural pipeline (cross-domain queries and informal queries) share a single shape: the first retrieval call does not have enough information to find every relevant chunk. Closing that gap calls either for a domain-adapted encoder, which trains the implicit memory into the network's weights, or for an explicit working memory that survives across multiple retrieval calls. The stateful agentic layer pursues the latter route.

Table 5.2 maps each component of the structural pipeline to the failure mode of the default pipeline that the component is intended to mitigate. The mapping is qualitative, not numerical, but it makes the design intent explicit and links it back to the limitations listed in Section 1.2.

A qualitative comparison with the prior retrieval-augmented system of [49] is informative but not quantitative. That system also pairs a dense retriever with a generator and produces a structured output, but the evaluation set, the metric definition, and the annotation authority differ, so a direct score comparison would be misleading. The methodological overlap is substantive: both systems treat the retriever as the dominant controller of the generator's behaviour. The framework in this thesis differs from [49] in its explicit handling of document structure during chunking, its rank-level fusion of two retrieval signals, its metadata-aware reranker, and its design for a stateful layer on top.

## 5.5 Limitations of the Evaluation

The evaluation has six limitations that bound the scope of the results reported above. The first four concern the corpus, the chunker, and the off-the-shelf retrievers; the last two concern the protocol that produces the headline accuracy figures.

**Corpus coverage.** The corpus contains forty central-level statutes. India's legislative corpus is larger by orders of magnitude. State laws, subordinate rules, amendments enacted after the snapshot date, and judicial decisions are not indexed. A complaint that hinges on a state law or on a recent amendment cannot be answered correctly even in principle.

**Chunking boundary conditions.** Sections longer than the 700-token target fall through to a paragraph or sentence separator. The resulting chunks are paragraph-level rather than section-level, which means the metadata at chunk level is less informative than for sections that fit inside the budget. A parent-child chunking scheme, in which the full section is stored as a parent and the paragraph-level pieces as searchable children, would resolve the issue without changing the rest of the pipeline. It is not implemented in the current system.

**Metadata coverage gap.** The 37.9% of chunks that lack a section number are ineligible for the provenance bonus regardless of how relevant they are to the query, and reranking cannot recover the gap on its own. A language-model-based fallback for unstructured headings, which would emit a section identifier for chunks that the regular-expression patterns miss, would substantially reduce this shortfall. It is also not in the current implementation.

**Retrieval domain transfer.** The dense encoder and the cross-encoder were trained on general-purpose paraphrase and web-search corpora. Query-document interaction patterns in the statutory domain differ from web search, and zero-shot transfer to out-of-domain collections is known to reduce nDCG@10 on most BEIR tasks relative to in-distribution numbers [8]. Domain-adaptive pre-training mitigates part of this drop in the European legal setting [9]; it is the closest documented remedy for the present pipeline in the literature cited here.

**Single-annotator ground truth.** Each of the two hundred complaints was annotated by one practising lawyer (Section 4.4). No second annotator scored the same complaints in parallel, so neither inter-annotator agreement nor an adjudication protocol for borderline cases is reported. The accuracy estimates and the Wilson intervals are therefore conditional on a single reference analysis. A two-annotator protocol with a disagreement-resolution step would tighten the bound on the headline accuracy figures and reduce the residual risk of label noise; it would also strengthen the empirical claim against label-noise objections in a straightforward way.

**Absence of comparative baselines and component ablations.** The evaluation reports the structural pipeline as a single configuration. It does not compare against the default RAG pipeline of Section 1.2, does not compare against a sparse-only or dense-only retriever, and does not ablate the provenance bonus, the MMR step, the cross-encoder, or the structure-aware chunker individually. The qualitative mapping in Table 5.2 is therefore design intent rather than measured contribution, and the discussion in Section 5.4 should be read as interpretive rather than as a causal attribution of the headline accuracy to any single component. Filling the gap with a factorial ablation on the same two-hundred-complaint benchmark is identified as follow-on work in Section 6.3.

The six limitations are independent. Closing any one of them would either improve the structural layer's score or tighten the empirical claim. None of them argues against the framework itself; each argues for a specific extension of it. Among the six, the protocol-side pair (single-annotator ground truth and the absence of comparative baselines and component ablations) most strongly bounds how the headline numbers can be interpreted, and they are addressed first in the future-work agenda of Section 6.3.

# CHAPTER 6

# CONCLUSION AND FUTURE WORK

The framework developed in this thesis treats contextual reasoning in a knowledge-intensive retrieval-augmented system as a pipeline design problem at least as much as a modelling problem. On the expert-annotated benchmark, the instantiated structural pipeline attains 94.0% primary-citation accuracy and 82.0% complete-citation accuracy without any domain-specific fine-tuning, by respecting document structure when chunking, carrying citation metadata through to the reranker, fusing dense and sparse retrieval at the rank level rather than the score level, and constraining the generator to a fixed output schema. The stateful agentic layer extends the same view to the time axis: when one retrieval call is not sufficient, a controller that maintains a working memory across calls and refines the query extends the same principle. This chapter draws the two layers together and sets out the next steps of the work.

## 6.1 Summary

The thesis sets out a framework for knowledge-intensive retrieval-augmented question answering. The framework has two layers. The structural layer addresses document hierarchy at every stage of the retrieve-rerank-generate pipeline: section-aware chunking, regular-expression metadata, dense and sparse retrievers running in parallel, Reciprocal Rank Fusion to merge their ranked lists, a cross-encoder reranker augmented with a metadata provenance bonus, Maximal Marginal Relevance for diversity, and a JSON schema enforced through a four-pass parser. The stateful agentic layer wraps a controller around the structural pipeline so that retrieval becomes one action inside a longer reasoning loop, with a persistent working memory, a reflect-and-refine step, and a confidence-gated stopping rule.

The empirical evidence comes from a single instantiation of the framework on a corpus chosen because it satisfies the four properties of a knowledge-intensive corpus identified in Chapter 1 (Section 1.1.1). The structural layer identifies the primary applicable citation in 94.0% of test queries (95% Wilson interval [89.8%, 96.5%]) and reproduces the full citation set in 82.0% of test queries (95% Wilson interval [76.1%, 86.7%]). The gap between the two numbers is the within-call versus across-call distinction the thesis began with, expressed as data: a single retrieval call is sufficient to identify a single primary citation, but not to recover the full set. Every neural component is off-the-shelf, so the reported numbers do not include any benefit that domain-specific fine-tuning of the encoder or reranker might add; the experiment does not, however, run a factorial ablation that would attribute the observed accuracy to individual architectural choices, and Section 6.3 identifies that ablation as the next empirical step.

## 6.2 Interpretation of the Reported Metrics

The 94.0% primary-citation accuracy indicates that, on the single-provision identification task, the combination of document-hierarchy chunking, hybrid retrieval, and metadata-augmented reranking reliably recovers the expert-designated primary citation on this benchmark; the absolute number alone does not establish superiority over unstated alternative pipelines without a controlled comparison. The 82.0% complete-citation accuracy shows that the same pipeline is

less adequate when the task is to recover the full set of applicable provisions, especially when the complaint crosses statutory domains or is expressed in informal vocabulary.

The twelve-point gap between the two scores measures the cost of single-pass retrieval on multi-statute complaints. The error analysis traces this cost back to three sources: cross-domain queries that stress dense retrieval across weakly related encoder neighbourhoods, informal queries that defeat the sparse retriever, and chunks without section-number metadata that the reranker cannot promote on content alone. None of the three is a property of the framework. Each is a property of the particular retrievers and the particular metadata coverage available at the time of the evaluation. The framework remains the same when each of them is addressed.

The empirical evaluation reported here applies to the structural layer only. The stateful agentic layer is a fully specified design (Section 3.3). Its empirical evaluation against the same benchmark is the principal next step. The thesis does not claim that the stateful layer improves on the structural layer's score; it claims that the framework admits both layers, that the interface between them is narrowly defined, and that the stateful layer is implementable on top of the structural layer without redesign.

## 6.3 Future Work

The directions below are listed in the order of expected impact on the metrics reported in Chapter 5.

**Empirical evaluation of the stateful agentic controller.** The highest-priority next step is to implement the controller of Section 3.3 on top of the existing structural pipeline and to evaluate the loop on the same two-hundred-complaint benchmark, with attention to the cross-domain and informal-language failure modes. The state representation, the reflect step, and the confidence-gated stopping rule are specified at a level that supports implementation, but the choice of language model for the controller, the cost of the reflection step at each iteration, and the iteration budget that balances quality against latency are open empirical questions. The confidence weights $(w_{\text{cov}}, w_{\text{agr}})$ introduced in Section 3.3.3 must also be fitted on a development split before the controller can be evaluated at scale.

**Comparative baselines and component ablations.** The limitations of Section 5.5 identified the absence of comparative baselines and of component ablations as the largest gap in the empirical evaluation. A factorial study on the same two-hundred-complaint benchmark, with each component of the structural pipeline (priority-ordered chunking, sparse retrieval, dense retrieval, rank fusion, provenance bonus, diversity selection) turned on and off independently, would convert the qualitative mapping in Table 5.2 into measured contribution and allow the framework to be compared against the default pipeline of Section 1.2 on a like-for-like basis.

**Annotator reliability and the size of the benchmark.** A second annotator on the same two-hundred complaints, with a documented adjudication protocol for disagreements, would convert the single-annotator estimate of Chapter 5 into a two-annotator estimate whose Wilson interval can be reported alongside an inter-annotator agreement statistic. The benchmark itself can also be enlarged by sourcing additional complaints that fall into the cross-domain and informal-language failure modes, so that the failure-mode counts of Section 5.3 stop being small-sample.

**Domain-adaptive encoders.** A sentence encoder fine-tuned on the target corpus is expected to substantially reduce the encoder-corpus mismatch identified in Section 2.4. Domain pre-training has been quantified for at least one specialist setting [9], and the same recipe (pre-train

on in-domain text, then fine-tune on a retrieval objective) applies to any corpus on which the framework is deployed. The structural layer is encoder-agnostic, so the substitution is mechanical once a domain encoder is available. The same argument applies to the cross-encoder, where a domain-adapted reranker would relax the constraint that the provenance bonus has to compensate for a generic model.

**Parent-child chunking and metadata fallbacks.** The two limitations of Sections 5.1.1 and 5.5 (sections longer than the budget; the 37.9% of chunks without section-number metadata) can be addressed by parent-child chunking and by a language-model-based metadata extractor for headings that the regular-expression patterns miss. Neither requires a change to the rest of the pipeline.

**Scaling to richly cross-referenced corpora.** Many knowledge-intensive corpora extend the textual content with a network of cross-references between documents. Case law refers to earlier cases; clinical guidelines refer to other guidelines; technical standards refer to other clauses. A graph-based retrieval layer on top of the existing index would expose these links to the retriever. Approaches that combine textual and network signals have been studied in the legal domain [47], and the broader survey of knowledge graphs in [51] reviews the relevant representation techniques. Cross-reference edges can also be embedded with relational methods such as TransE [52] to give the retriever a graph-aware signal alongside the textual one. As the corpus grows, the dense index will eventually exceed the size at which exact inner-product search is the right choice; the framework already supports the substitution of HNSW [43] in place of the flat index, with no other change to the pipeline.

**Multilingual extension.** The corpus and the benchmark used in this evaluation are in English. Many knowledge-intensive domains are inherently multilingual; in those settings, parallel corpora are not always available, and retrieval requires either a multilingual sentence encoder or document-side translation at index time. The structural layer can absorb a multilingual encoder, but the provenance metadata and the output schema have to be re-keyed across languages. The cross-lingual case is the most resource-intensive of the directions listed above and depends on prior corpus-level work.

# BIBLIOGRAPHY

[1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020, pp. 9459–9474.

[2] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown, and Y. Shoham, "In-context retrieval-augmented language models," *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 1316–1331, 2023.

[3] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the middle: How language models use long contexts," *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 157–173, 2024.

[4] H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal, "Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions," in *Proc. 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023, pp. 10 014–10 037.

[5] Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, "Active retrieval augmented generation," in *Proc. 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023, pp. 7969–7992.

[6] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-RAG: Learning to retrieve, generate, and critique through self-reflection," in *Proc. 12th International Conference on Learning Representations (ICLR)*, 2024.

[7] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," in *Proc. 11th International Conference on Learning Representations (ICLR)*, 2023.

[8] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, "BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models," in *Proc. 35th Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

[9] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "LEGAL-BERT: The muppets straight out of law school," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 2898–2904.

[10] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," *Foundations and Trends in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.

[11] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, S. Gupta, B. P. Majumder, K. Hermann, S. Welleck, A. Yazdanbakhsh, and P. Clark, "Self-refine: Iterative refinement with self-feedback," in *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[12] N. Shinn, F. Cassano, E. Berman, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," in *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[13] G. V. Cormack, C. L. A. Clarke, and S. Buettcher, "Reciprocal rank fusion outperforms Condorcet and individual rank learning methods," in *Proc. 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2009, pp. 758–759.

[14] J. Carbonell and J. Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in *Proc. 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998, pp. 335–336.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, 2017, pp. 5998–6008.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019, pp. 4171–4186.

[17] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[18] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020, pp. 1877–1901.

[19] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proc. 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020, pp. 7871–7880.

[20] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang, "REALM: Retrieval-augmented language model pre-training," in *Proc. 37th International Conference on Machine Learning (ICML)*, 2020, pp. 3929–3938.

[21] G. Izacard and E. Grave, "Leveraging passage retrieval with generative models for open domain question answering," in *Proc. 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021, pp. 874–880.

[22] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. van den Driessche, J.-B. Lespiau, B. Damoc, A. Clark *et al.*, "Improving language models by retrieving from trillions of tokens," in *Proc. 39th International Conference on Machine Learning (ICML)*, 2022, pp. 2206–2240.

[23] K. Shuster, S. Poff, M. Chen, D. Kiela, and J. Weston, "Retrieval augmentation reduces hallucination in conversation," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021, pp. 3784–3803.

[24] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald, "On faithfulness and factuality in abstractive summarization," in *Proc. 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020, pp. 1906–1919.

[25] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, 2023.

[26] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022, pp. 24 824–24 837.

[27] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," in *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[28] L. Gao, X. Ma, J. Lin, and J. Callan, "Precise zero-shot dense retrieval without relevance labels," in *Proc. 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023, pp. 1762–1777.

[29] J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, "Generative agents: Interactive simulacra of human behavior," in *Proc. 36th Annual ACM Symposium on User Interface Software and Technology (UIST)*, 2023.

[30] D. C. Blair and M. E. Maron, "An evaluation of retrieval effectiveness for a full-text document-retrieval system," *Communications of the ACM*, vol. 28, no. 3, pp. 289–299, 1985.

[31] H. Turtle and W. B. Croft, "Evaluation of an inference network-based retrieval model," *ACM Transactions on Information Systems*, vol. 9, no. 3, pp. 187–222, 1991.

[32] J. M. Ponte and W. B. Croft, "A language modeling approach to information retrieval," in *Proc. 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998, pp. 275–281.

[33] C. Zhai and J. Lafferty, "A study of smoothing methods for language models applied to information retrieval," *ACM Transactions on Information Systems*, vol. 22, no. 2, pp. 179–214, 2004.

[34] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[35] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3982–3992.

[36] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. tau Yih, "Dense passage retrieval for open-domain question answering," in *Proc. 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 6769–6781.

[37] L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. N. Bennett, J. Ahmed, and A. Overwijk, "Approximate nearest neighbor negative contrastive learning for dense text retrieval," in *Proc. 9th International Conference on Learning Representations (ICLR)*, 2021.

[38] S. Hofstätter, S.-C. Lin, J.-H. Yang, J. Lin, and A. Hanbury, "Efficiently teaching an effective dense retriever with balanced topic aware sampling," in *Proc. 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 113–122.

[39] S.-C. Lin, J.-H. Yang, and J. Lin, "In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval," in *Proc. 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, 2021, pp. 163–173.

[40] O. Khattab and M. Zaharia, "ColBERT: Efficient and effective passage search via contextualized late interaction over BERT," in *Proc. 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 39–48.

[41] J. Lin, R. Nogueira, and A. Yates, *Pretrained Transformers for Text Ranking: BERT and Beyond*, ser. Synthesis Lectures on Human Language Technologies. Morgan & Claypool, 2021.

[42] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, 2019.

[43] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 4, pp. 824–836, 2020.

[44] H. Zhong, C. Xiao, C. Tu, T. Zhang, Z. Liu, and M. Sun, "How does NLP benefit legal system: A summary of legal artificial intelligence," in *Proc. 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020, pp. 5218–5230.

[45] I. Chalkidis, A. Jana, D. Hartung, M. Bommarito, I. Androutsopoulos, D. M. Katz, and N. Aletras, "LexGLUE: A benchmark dataset for legal language understanding in English," in *Proc. 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022, pp. 4310–4330.

[46] N. Aletras, D. Tsarapatsanis, D. Preoţiuc-Pietro, and V. Lampos, "Predicting judicial decisions of the European Court of Human Rights: A natural language processing perspective," *PeerJ Computer Science*, vol. 2, p. e93, 2016.

[47] P. Bhattacharya, K. Ghosh, A. Pal, and S. Ghosh, "Legal case document similarity: You need both network and text," *Information Processing & Management*, vol. 59, no. 6, p. 103069, 2022.

[48] D. Locke, G. Zuccon, and H. Scells, "Automatic query generation from legal texts for case law retrieval," in *Information Retrieval Technology: 13th Asia Information Retrieval Societies Conference (AIRS 2017)*, ser. Lecture Notes in Computer Science. Springer, 2017, vol. 10648, pp. 181–193.

[49] F. Mubeen, A. Mehdi, M. A. Haque, M. Z. M. Nomani, and N. S. Uddin, "Redefining legal access: a RAG-based AI system for Indian law," *Human-Intelligent Systems Integration*, vol. 7, pp. 87–98, 2025.

[50] E. B. Wilson, "Probable inference, the law of succession, and statistical inference," *Journal of the American Statistical Association*, vol. 22, no. 158, pp. 209–212, 1927.

[51] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A.-C. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, and A. Zimmermann, "Knowledge graphs," *ACM Computing Surveys*, vol. 54, no. 4, pp. 71:1–71:37, 2021.

[52] A. Bordes, N. Usunier, A. Garcia-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, 2013, pp. 2787–2795.