LogicTree-RAG: Logic Tree-guided Retrieval-Augmented Generation for Long-form Patent Drafting

Jiaqi Zhu
National University of Singapore
jiaqi77@nus.edu.sg

Naili Xing *
National University of Singapore
xingnl@comp.nus.edu.sg

Hexiang Pan
National University of Singapore
panh@u.nus.edu

Haotian Gao
National University of Singapore
gaohaotian@comp.nus.edu.sg

Jianwei Yin
Zhejiang University
zjuyjw@zju.edu.cn

Xiaokui Xiao
National University of Singapore
xkxiao@nus.edu.sg

Beng Chin Ooi
Zhejiang University
ooibc@zju.edu.cn

Abstract

Long-form technical text generation underpins knowledge-intensive workflows, yet remains challenging for large language models (LLMs) due to the need for globally consistent logical structuring and faithful technical reasoning beyond local coherence. Patent drafting is a canonical instance of this challenge, demanding holistic generation of a legally compliant and technically exhaustive document through sustained multi-expert collaboration. Existing approaches often focus on partial section generation or rely on manually crafted outlines, limiting scalable automation in realistic settings. In this work, we propose LogicTree-RAG, a logic tree-guided retrieval-augmented generation framework that induces a hierarchical logic tree as a global organizational backbone to organize and ground technical disclosures, without relying on expert-defined drafting priors. Each node in the logic tree represents a technical element and is constructed through evidence-guided recursive generation. A hybrid traversal mechanism then maps the logic tree into patent sections, enabling controllable and section-balanced generation. Extensive experiments show that LogicTree-RAG consistently improves content quality and language conformity over strong LLM-based baselines and achieves longer structured generation with high token efficiency, demonstrating the effectiveness of logic-centric generation for complex technical document drafting.

1 Introduction

Patents are a cornerstone of intellectual property protection, safeguarding technological innovations and enabling their translation into practical applications [1, 2]. Drafting a patent, however, is a highly specialized task that requires both a comprehensive understanding of the underlying technical contribution and a detailed knowledge of the formal and legal conventions of patent documents [3, 4]. In practice, this process often involves close collaboration between *inventors*, who possess the technical expertise but typically lack familiarity with patent law, and *patent attorneys*, who are well-versed in legal conventions but may not fully understand the technical intricacies of the invention. Consequently, such multi-expert collaboration often necessitates repeated discussion and iterative refinement, making the drafting process labor-intensive and time-consuming.

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Recent advances in large language models (LLMs), which have demonstrated remarkable performance across a wide range of natural language processing (NLP) tasks, open up new possibilities for automating and accelerating patent drafting [5, 6, 7]. Prior work in this area has primarily focused on generating specific sections of patent applications [8, 9, 10, 11, 12, 13, 14], such as title, abstract, or claims, rather than producing a complete document. While useful for targeted drafting assistance, such partial generation fails to address the core challenge of producing a logically structured patent application in its entirety, where interdependencies across sections are essential.

While a few recent studies have explored full-patent generation, they typically rely on substantial manual pre-design and drafting efforts. For instance, AutoPatent [15] presupposes user-provided draft information, while Pap2Pat [16] assumes a detailed outline of each patent section written in advance. In their task setups, both methods approximate such user-provided inputs by extracting supervision from reference patents, which distances their designs from real-world patent drafting conditions and ultimately limits automation and scalability.

To address the above-mentioned problems, we formulate patent drafting as a long-text generation task, focusing on producing a complete and logically structured patent application from a pre-publication research paper serving as a proxy for the invention report (IR) [16]. This formulation captures the practical scenario in many research environments, where research papers frequently serve as the initial disclosure for patent drafting. Despite its practical relevance, advancing toward fully automated patent drafting entails several non-trivial challenges. First, patent applications are inherently long-form technical documents, requiring the generation of substantially long and complete texts that remain challenging for LLMs to sustain [17, 18]. Second, beyond producing long outputs, effective patent drafting demands globally consistent logical organization and balanced elaboration, a constraint that becomes increasingly difficult to maintain as generation length grows. Third, robust automation requires minimizing dependence on manually crafted priors, as such reliance limits scalability and generalization across drafting scenarios.

To tackle these challenges, we propose LogicTree-RAG, a logic tree-guided retrieval-augmented generation framework for long-form patent drafting. Unlike prior methods that depend on flat-generation LLMs with manually crafted outlines or templates, LogicTree-RAG introduces a structured generative paradigm in which *logic precedes content*: it automatically derives a hierarchical logic tree from the input research paper as a global organizational backbone and progressively transforms it into well-formed patent sections in a principled and controllable manner.

As the first step, LogicTree-RAG incorporates a semantic chunking mechanism that structures the input document into semantically coherent units, which are embedded into a vectorized document knowledge base. Upon this foundation, we introduce an evidence-guided recursive generation process for constructing a logic tree that captures the conceptual structure of the underlying research. This process leverages a novel node-aware semantic search strategy equipped with complementary expansion and refinement procedures, enabling section-level coherence and structural balance. Each node in the tree corresponds to a technical element, such as a principle, subtask, or subsystem, and is dynamically grounded by evidence retrieved from the knowledge base. Finally, a hybrid traversal strategy transforms the completed logic tree into patent sections, yielding a long-form, logically organized patent draft without requiring expert-defined priors. By serving as a logic-centric intermediate representation, logic tree bridges factual evidence with global generative control, enabling scalable long-form patent drafting.

We summarize our main contributions as follows:

- We propose LogicTree-RAG, a logic-centric generative framework for research-to-patent transformation that produces long-form, logically structured patent drafts without reliance on expert-defined drafting priors.
- We present an evidence-guided recursive generation approach that constructs a logic tree to model the conceptual dependencies and technical contributions with explicit factual grounding in the source document.
- We introduce a node-aware semantic search with a discriminative relevance scoring that jointly accounts for semantic similarity and structural distinctiveness, improving balanced elaboration and global structural coherence in patent drafting.
- Extensive experiments under a curated suite of evaluation metrics demonstrate that LogicTree-RAG improves technical completeness by 11.03% and content fidelity by 6.33% over strong LLM-based baselines, while enabling longer structured generation with high token efficiency.

2 Preliminaries

Patents and Patent-Paper Pairs. A patent is a structured legal-technical document, typically consisting of *claims*, which define the legally enforceable scope of protection, and a *description*, which discloses the invention in sections such as Background, Summary, and Detailed Description [19, 16]. In many research domains, patents are filed alongside scientific publications, producing *Patent-Paper Pairs* [20, 21, 22, 23], where a pre-publication paper serves as the IR for the patent. Patent-paper pairs align scientific discourse with legal–technical documentation, offering a valuable yet underexplored resource for studying long-form technical text generation with LLMs.

Problem Formulation. Consider a patent-paper pair (X, P), where $X = \{x_1, x_2, \ldots, x_m\}$ is a pre-publication research paper represented as a collection of text spans, and $P = \{p_1, p_2, \ldots, p_n\}$ is the corresponding patent document. In this work, we formalize the task as generating patent P from paper X mediated by a logic tree $\mathcal{T}$ with evidence grounding, defined as follows.

Definition 2.1 (Logic Tree)**.** A logic tree is a directed rooted tree $\mathcal{T} = (V, E, r)$, where $V = \{v_1, \cdots, v_N\}$ is the set of nodes, $E \subseteq V \times V$ the set of directed edges specifying parent-child relations, and $r \in V$ the root node representing the overall description of the document. Each node $v \in V$ is associated with two attributes: (i) a generated text fragment $c(v)$ and (ii) a set of supporting evidence spans $\mathcal{E}(v) \subseteq X$.

Intuitively, the root node summarizes the invention, internal nodes correspond to higher-level sections, and leaf nodes expand into fine-grained technical details. To ensure factual grounding, we impose the attribution constraint

$$\forall v \in V, \quad \mathcal{E}(v) \neq \emptyset, \tag{1}$$

which requires each generated unit to be anchored to at least one evidence span in the source paper.

3 Methodology

3.1 LogicTree-RAG Overview

As illustrated in Figure 1, LogicTree-RAG functions as a top-down pipeline that first parses the source document into structured textual units and constructs a logic tree that captures the conceptual dependencies. Each node in the tree represents a technical element, such as a principle, step, or subsystem, and is dynamically associated with retrieved evidence from a vectorized document knowledge base. LogicTree-RAG then iteratively expands and refines the tree via evidence-guided recursive generation, supported by node-aware semantic search to ensure precise evidence alignment and structural consistency. Finally, the framework traverses the completed logic tree and hierarchically aggregates node contents into patent sections, yielding a long-form, section-balanced patent draft that integrates technical fidelity with structured generative controllability.

3.2 Hierarchical Logic Tree Initialization

To initialize retrievable knowledge units, we utilize layout-aware document analysis [24] to identify hierarchical boundaries across sections and paragraphs, reconstructing the document's logical structure. Subsequently, we perform semantic chunking to produce contextually coherent segments that from the basis of the vectorized document knowledge base $\mathcal{V}_{kb}$. Unlike fixed chunking methods [25, 26, 27, 16], our approach dynamically determines chunk boundaries according to the semantic coherence and contextual dependency between adjacent paragraphs. Formally, given an ordered sequence of paragraphs $X = \{x_1, x_2, \ldots, x_m\}$, we compute the pairwise semantic similarity:

$$s_{i,i+1} = \text{sim}(\mathcal{E}_p(x_i), \mathcal{E}_p(x_{i+1})),\ i = 1, \cdots, m-1, \tag{2}$$

where $\mathcal{E}_p(\cdot)$ denotes the paragraph-level embedding function, and $\text{sim}(\cdot, \cdot)$ is instantiated with cosine similarity. Paragraphs are iteratively merged when $s_{i,i+1} \geqslant \tau_s$ and the accumulated token length does not exceed the upper bound t_{max}. This procedure yields a set of semantically coherent chunks $\tilde{X} = \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_L\}$ that align with the recovered hierarchy. Details are provided in Appendix C.

Following semantic chunking, we initialize a logic tree $\mathcal{T} = (V, E, r)$ to organize and drive the generation process. To construct an initial structural scaffold, we prompt the backbone LLM with a high-level instruction σ_I and the chunk set $\tilde{X}$, generating a draft technical description

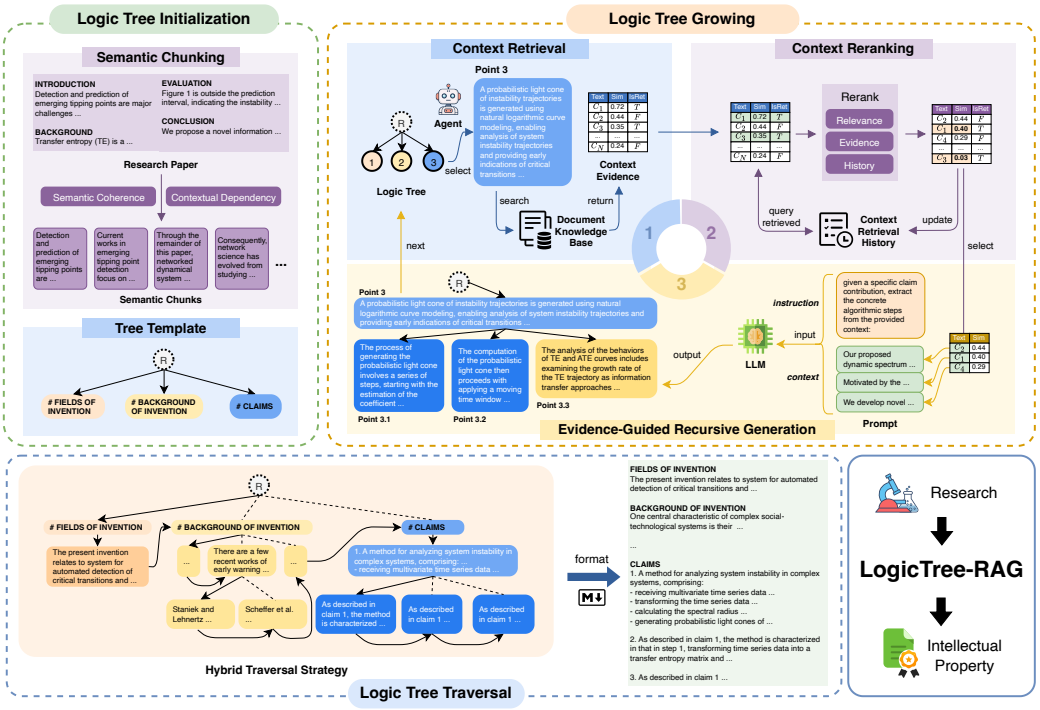


Figure 1: Architecture of LogicTree-RAG framework for long-form patent drafting.

$\mathcal{D}_d = \mathcal{M}_{LLM}(\sigma_I, \tilde{X})$. The root node r is instantiated with $c(r) = \mathcal{D}_d$ and $\mathcal{E}(r) = \tilde{X}$, representing the target patent document. Instead of utilizing syntactic rules or heuristics, we leverage $\mathcal{M}_{LLM}$ to induce the first layer of the logic tree by producing an evidence-grounded decomposition of key technical components from $\mathcal{D}_d$. This yields an ordered tree structure in which the root summarizes the invention and its first-level children nodes define the core semantic dimensions to be recursively expanded in subsequent stages.

3.3 Evidence-Guided Recursive Generation

Given the initialized logic tree $\mathcal{T}$, we perform recursive generation through a breadth-first expansion strategy. Specifically, we maintain a FIFO search queue $\mathcal{Q}$ initialized with the leaf nodes of $\mathcal{T}$, ensuring the expansion proceeds in a top–down manner. For each iteration, we dequeue the front node v_t, and retrieve its parent node v_f from $\mathcal{T}$. The goal of recursive generation is to determine whether v_t should be expanded, refined, or skipped based on the semantic evidence retrieved from the document knowledge base $\mathcal{V}_{kb}$, and to perform the corresponding updates to the logic tree.

To obtain candidate evidence, we perform retrieval over $\mathcal{V}_{kb}$ and collect a set of context spans $\mathcal{D}_{ctx}$, referred to as context evidence, which are semantically aligned with the node's content $c(v_t)$. A global novelty test is then designed to verify whether $\mathcal{D}_{ctx}$ is already covered by other nodes in the logic tree, excluding the child nodes of v_f. If no novel evidence is identified, v_t is skipped; otherwise, the logic tree is updated via two complementary procedures:

- Expansion procedure $E(\cdot)$: Conditioned on $\mathcal{D}_{ctx}$ and an expansion instruction σ_E, the model generates a supplementary paragraph, which is then segmented into coherent semantic units. Each unit is instantiated as a child node of v_t, forming the set $V_t^{child} = \{v_{tc1}, \cdots, v_{tck}\}$, where $c(v_{tci})$ is initialized as the corresponding segmented content and $\mathcal{E}(v_{tci}) = \mathcal{D}_{ctx}$. All nodes in V_t^{child} are then enqueued into $\mathcal{Q}$ for further recursion.
- Refinement procedure $R(\cdot)$: The model examines $c(v_t)$ to identify under-specified terms using instruction σ_{R1}. For domain-specific concepts appearing in $\tilde{X}$, refinement is performed via retrieval-augmented generation with supporting evidence from $\mathcal{V}_{kb}$ and instruction σ_{R2}; otherwise, the model completes the term with instruction σ_{R3} based on its internal knowledge.

Overall, the logic tree $\mathcal{T}$ is updated after each expansion or refinement step. The expansion procedure generates a set of child nodes based on the context evidence, while the refinement procedure resolves under-specified terms and updates the content of the current node, expressed as:

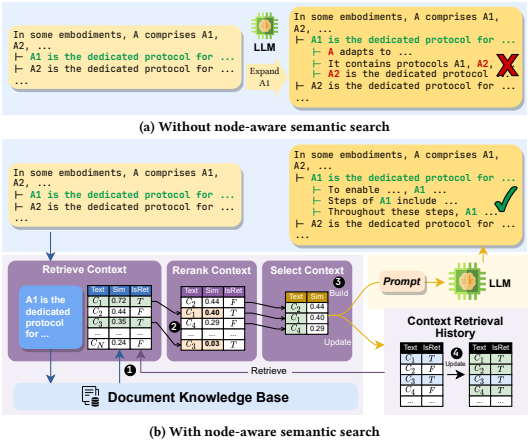
$$E(\sigma_E, \mathcal{D}_{ctx}) = V_t^{child} \tag{3}$$

$$c(v_t) \leftarrow R(\sigma_R, c(v_t), \mathcal{V}_{kb}) \tag{4}$$

where $\sigma_R = \{\sigma_{R1}, \sigma_{R2}, \sigma_{R3}\}$. The recursive process terminates once either the queue becomes empty or the accumulated content length reaches a predefined budget, ensuring controllable expansion and preventing over-generation.

3.4 Node-Aware Semantic Search

During the evidence-guided recursive generation, the quality of the context evidence $\mathcal{D}_{ctx}$ critically influences the structural evolution of the logic tree. Naive retrieval may introduce redundancy or topic drift, particularly when the retrieved spans overlap with already-processed sibling nodes, leading to incoherent or uneven expansion. To ensure precision and structural consistency, we impose two desiderata for the context set: (i) strong semantic association with the current node, and (ii) low redundancy with its sibling nodes under the same parent. To meet these requirements, we introduce a node-aware semantic search mechanism. As illustrated in Figure 2, we first retrieve an initial set of candidate context spans $\tilde{\mathcal{D}}_{ctx}$ from $\mathcal{V}_{kb}$ using the current node content $c(v_t)$ as query:



(a) Without node-aware semantic search

(b) With node-aware semantic search

Figure 2: Generation without and with node-aware semantic search.

$$\tilde{\mathcal{D}}_{ctx} = \mathcal{R}(c(v_t), \mathcal{V}_{kb}) = \text{Top-K}_{a \in \mathcal{V}_{kb}} \text{sim}(a, c(v_t)), \tag{5}$$

To jointly account for semantic specificity and structural discrimination, we evaluate each candidate span $d \in \tilde{\mathcal{D}}_{ctx}$ through a discriminative relevance score $S_{dr}(\cdot)$, defined as:

$$S_{dr}(d) = \text{sim}(d, c(v_t)) - \max_{v_e \in V_f^{child} \setminus \{v_t\}} \text{sim}(d, c(v_e)), \tag{6}$$

where V_f^{child} denotes the set of child nodes of v_f and $\setminus$ indicates exclusion. A span d is retained only if $S_{dr}(d)$ exceeds a predefined threshold. The final context evidence is obtained as $\mathcal{D}_{ctx} = \{d \,|\, S_{dr}(d) \geq \tau_f\}$. The node-aware semantic search ensures that the retrieved evidence is both node-specific and structurally non-redundant, supporting focused and coherent recursive generation.

Remark. Although both the node-aware semantic search and the global novelty test operate on evidence relevance and redundancy, they address complementary levels of semantic discrimination. Node-aware semantic search enforces specificity at the *local sibling level* by filtering evidence overlapping with existing child nodes of v_t, while the global novelty test verifies coverage at the *global tree-wide level* by checking whether $\mathcal{D}_{ctx}$ has already appeared elsewhere in the logic tree outside the v_f branch. Together, these mechanisms provide coordinated local discrimination and global redundancy control, enabling coherent and non-repetitive logic tree construction.

3.5 LogicTree-RAG Workflow for Patent Generation

As outlined in Algorithm 1, LogicTree-RAG operates in two stages: (i) *logic tree construction*, which provides a structured and factually grounded representation of the source document's conceptual space, and (ii) *patent drafting*, which systematically traverses the constructed logic tree to generate the final patent draft. The first stage yields a logic tree $\mathcal{T}$ that associates each node $v \in V$ with both generated content $c(v)$ and grounded evidence $\mathcal{E}(v) \subseteq X$, as formalized in Algorithm 2 in Appendix D, forming the structural foundation for patent drafting.

In the second stage, LogicTree-RAG generates the patent draft by traversing the logic tree $\mathcal{T}$ and aggregating node contents. The hierarchical structure of the logic tree naturally aligns with the

Algorithm 1 LogicTree-RAG for Patent Generation

Input: Pre-publication research paper, the backbone model $\mathcal{M}_{LLM}$, instruction prompts $\sigma_I, \sigma_E, \sigma_R$.
Output: Generated patent document P.

Stage I: Logic Tree Construction

Perform layout-aware semantic chunking on X to obtain chunk set $\tilde{X}$ using Eq. (2)
Initialize Logic Tree $\mathcal{T}$ by generating a draft technical description $\mathcal{D}_d = \mathcal{M}_{LLM}(\sigma_I, \tilde{X})$
Expand and refine $\mathcal{T}$ via evidence-guided recursive generation with σ_E, σ_R as Eq. (3) and Eq. (4)

Stage II: Patent Drafting

Initialize empty patent document P
for section $s \in \{\texttt{Background}, \texttt{Summary}\}$ **do**
    Traverse $\mathcal{T}$ in BFS order over nodes assigned to section s
    Generate section text conditioned on the content $c(v)$ of visited nodes using $\mathcal{M}_{LLM}$
    Append generated content to P
end for
Identify subtrees $\{\mathcal{T}_d\}$ allocated to the `Detailed Description` section
for each subtree $\mathcal{T}_d$ **do**
    Traverse $\mathcal{T}_d$ in DFS order
    Generate fine-grained technical content using $\mathcal{M}_{LLM}$
    Append generated content to P
end for
Return Generated patent document P

organization of complex technical documents, such as patents, where high-level nodes capture overarching concepts, and deeper nodes progressively elaborate technical details, procedural logic, and implementation specifics. To effectively exploit this hierarchical representation, we design a hybrid traversal strategy that balances content coverage and technical depth across patent sections. Specifically, high-level sections such as Background and Summary are generated using breadth-first traversal (BFS) to ensure comprehensive and balanced structural coverage. For the Detailed Description section, BFS is first combined with LLM-based semantic clustering to identify coherent logical subtrees, each of which is then expanded via depth-first traversal (DFS) to elaborate technical details. In this way, BFS establishes the structural skeleton, while DFS enables fine-grained expansion, jointly ensuring structural clarity and semantic specificity. Ultimately, LogicTree-RAG leverages the logic tree as a structural backbone to automatically transform research documents into long-form patent drafts without relying on manually crafted templates or outlines derived from ground truth.

Remark. LogicTree-RAG enables scalable and controllable long-form generation through an explicit hierarchical intermediate representation. By decomposing the source document into a logic tree and constraining generation to structurally relevant subtrees, the computational cost is bounded by the tree size and traversal budget, enabling fine-grained control over content expansion. Moreover, by localizing evidence retrieval via node-aware semantic search, LogicTree-RAG reduces redundant context consumption and improves token efficiency for long and complex technical documents.

4 Experiments

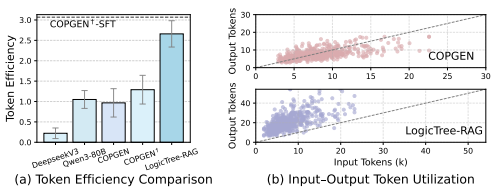
4.1 Experimental Setup

Benchmarks. We first conduct experiments on Pap2Pat [16], a recently proposed benchmark for long-form patent generation based on paper-patent pairs. Pap2Pat consists of 1813 aligned scientific papers and their corresponding granted patents across various domains. To further evaluate the generalizability of our framework beyond patent drafting, we additionally consider the LCFO benchmark [28], which focuses on long-context summarization and document expansion. LCFO consists of 252 long documents ($\sim$5k words each) spanning multiple domains, each paired with multi-level summaries and QA annotations, enabling controlled evaluation of long-form expansion.

Baselines. We compare LogicTree-RAG with 13 baselines across three categories: (i) *Heuristic Skyline*, including ground-truth patents as an oracle upper bound and a case where the source paper is treated as the generated document [16]. (ii) *Single LLM-Call*, where open and proprietary long-context LLMs (Kimi K2 [29], Qwen3-80B, Qwen3-Max [30], DeepseekV3 [31], and GPT-5 [32] and Gemini-2.5 Pro [33] under deep thinking mode) are prompted to generate patents from the full paper. (iii) *Agentic Framework*, including COPGEN [16], along with its SFT-tuned variant, LongWriter [34], and LongWriter-Zero [35], as well as agentic paradigms ReAct [36] and Plan-then-Execute [37] for step-wise planning and tool use. See Appendix F.2 for baselines and implementation details.

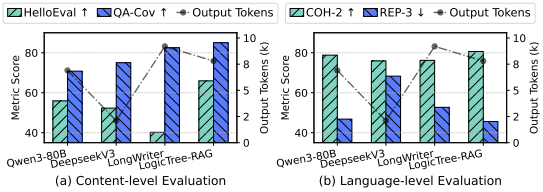
Table 1: Overall performance comparison of long-form patent drafting. The best results are highlighted in **bold**, and the second-best results are in **bold**.

Method	Output Tokens	Content-level				Language-level			Time (s) / Patent (Token)
		Coverage ↑	Factuality ↑		Semantic Similarity ↑	Style ↑	Repetition $\mathcal{R}_r$ ↓	Coherence ↑	
			$\mathcal{F}_{Pat}$	$\mathcal{F}_{Src}$					
Heuristic Skyline									
Reference Patent (GT)	18.16k	89.05	88.74	88.80	100	100	13.91	100	-
Source Paper	8.03k	44.77	46.06	88.68	39.37	39.26	8.39	98.60	-
Single LLM-Call Generation									
Kimi K2	3.26k	38.21	42.76	58.95	21.28	39.47	12.04	95.92	64 (19.63)
Qwen3-80B	8.91k	39.09	37.50	44.88	34.08	35.06	6.05	95.71	81 (9.10)
Qwen3-Max	8.91k	39.44	37.68	45.22	34.09	35.06	8.41	97.14	172 (19.29)
DeepseekV3	1.87k	36.70	47.09	59.66	11.98	32.68	6.13	95.56	43 (22.89)
GPT-5 DT	2.95k	40.91	46.70	62.04	20.56	41.52	7.25	97.34	146 (49.49)
Gemini-2.5 Pro DT	2.69k	38.69	47.58	60.97	24.25	39.68	6.09	97.12	109 (40.52)
Agentic Framework Generation									
COPGEN	7.69k	39.21	57.63	62.39	32.56	59.54	15.47	97.11	265 (34.46)
COPGEN†	8.96k	32.85	52.44	58.16	28.96	43.38	18.94	97.21	216 (24.11)
↪w/ SFT	25.85k	40.77	50.12	59.69	38.72	62.21	21.58	**97.50**	221 (8.55)
LongWriter	10.81k	37.21	36.65	45.21	21.12	34.26	6.67	95.62	197 (18.22)
LongWriter-Zero	12.16k	36.92	38.36	46.17	22.86	36.72	5.76	96.84	258 (21.11)
LogicTree-RAG	20.22k	**43.79**	**58.92**	**66.34**	**38.91**	**65.68**	**4.14**	**97.45**	311 (15.38)
↪Contamination Control	18.96k	**42.95**	**57.79**	**66.57**	**40.28**	**64.28**	**4.60**	97.23	298 (15.72)



(a) Token Efficiency Comparison (b) Input–Output Token Utilization

Figure 3: Token efficiency comparison results.



(a) Content-level Evaluation (b) Language-level Evaluation

Figure 4: Generalization performance on LCFO.

Implementation. LogicTree-RAG is implemented in Python with Pytorch 2.0.1, with Milvus v2.4.7 as the document knowledge base for document indexing and retrieval. All methods are evaluated under identical LLM decoding configurations to ensure a fair comparison. Detailed prompts and implementation settings are provided in Appendices E and F.

Evaluation Metrics. We evaluate patent generation from both content-level and language-level perspectives, complemented by a human expert study conducted by patent practitioners. *Content-level* metrics assess whether the generated patent conveys accurate and sufficient technical information grounded in the reference materials, including Coverage (NLI-based SCALE [38]), Factuality (patent-grounded $\mathcal{F}_{Pat}$ and source-grounded $\mathcal{F}_{Src}$), and Semantic Similarity (ROUGE-L [39]). *Language-level* metrics evaluate adherence to patent writing conventions, including Style (StyloMetrix [40]), Repetition ($\mathcal{R}_r$ [41]), and Coherence (DiscoScore [42]). Details are provided in Appendix F.1.

4.2 Main Results

The overall results are summarized in Table 1. LogicTree-RAG delivers consistently superior performance across key content and language metrics, generating substantially longer patent drafts that outperform strong long-context baselines. At the *content level*, LogicTree-RAG achieves the highest coverage, factuality, and semantic similarity among all baselines. These results underscore that logic-tree-guided generation improves technical completeness and evidence-grounded elaboration beyond single-call long-context prompting and agentic frameworks even with post-training. At the *language level*, LogicTree-RAG demonstrates strong performance in stylistic conformity and repetition rate. Its coherence score ranks second only to the SFT-tuned COPGEN variant, while requiring no task-specific fine-tuning. To control for potential training-data contamination, we evaluate LogicTree-RAG on a temporally filtered post-2024 subset beyond typical pretraining cutoffs, observing consistent gains without degradation, suggesting that the improvements are not driven by training-data overlap. Moreover, LogicTree-RAG maintains competitive per-token efficiency with only modest overhead, offering a favorable quality–efficiency trade-off for long-form patent drafting.

4.3 In-Depth Analysis of Patent Generation

Token Efficiency. We further analyze token efficiency, defined as the ratio of generated output tokens to total input tokens consumed across all LLM invocations, which serves as a proxy for both

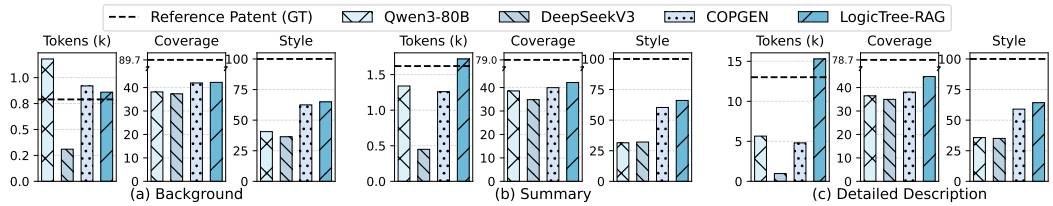


Figure 5: Section-level performance in terms of output length, content coverage, and linguistic style.

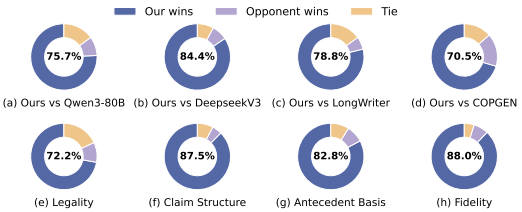


Figure 6: Win-rate results (%) under pairwise comparison.

Table 2: Component-wise ablation of LogicTree-RAG.

Variant	Output Tokens	Content-level		Language-level	
		Coverage	Factuality ($\mathcal{F}_{Pat}$)	Style	RR ($\mathcal{R}_r$)
w/o Logic Tree	9.10k	39.85	56.92	60.47	16.86
Logic Tree w/o Retrieval	8.21k	40.56	51.16	59.05	9.38
w/o Semantic Chunking	19.42k	42.09	53.74	62.27	10.93
w/o Refinement	18.58k	39.92	56.85	61.87	**4.09**
w/o Evidence-Grounded σ_{R2}	19.60k	40.16	56.98	62.64	4.67
w/o Internal Knowledge σ_{R3}	18.95k	43.58	58.85	65.62	5.21
w/o Node-aware Reranking	20.85k	42.60	57.13	64.79	8.32
w/ DFS Traversal	21.13k	41.15	58.68	65.12	5.34
LogicTree-RAG	20.22k	**43.79**	**58.92**	**65.68**	4.14

generation efficiency and scalability. As illustrated in Figure 3(a), LogicTree-RAG achieves the highest token efficiency among all baselines, closely approaching the SFT-enhanced COPGEN variant despite requiring no task-specific fine-tuning. Figure 3(b) further shows that LogicTree-RAG shifts the input–output token distribution toward a more efficient regime and effectively converts contextual input into useful long-form patent content. Detailed cost breakdowns are provided in Appendix G.4.

Generalizability to Long-Text Generation. LogicTree-RAG is not tied to patent drafting, but targets a broader setting of source-grounded structured long-form generation, where the goal is to transform a long source document into a coherent and factually grounded target document. To validate this generalizability, we evaluate LogicTree-RAG on the LCFO benchmark [28] for document expansion. Following [28], we report output tokens, coherence (COH-2), repetition (REP-3), overall quality via an LLM-as-a-judge metric (HelloEval), and human-evaluated QA-based coverage (QA-Cov), which measures the fraction of benchmark questions answerable from the generated output. Without modifying the core pipeline, we apply LogicTree-RAG by adapting only the output schema. As shown in Figure 4, LogicTree-RAG consistently outperforms baselines in all content and language-level metrics, while maintaining competitive output length. These findings suggest that the benefits of logic-tree-guided generation extend beyond patent drafting to general long-form generation tasks.

Section-Level Analysis. We analyze generation quality across patent sections, i.e., Background, Summary, and Detailed Description, considering length, content coverage, and linguistic style, with results summarized in Figure 5. LogicTree-RAG produces section-wise output lengths closely aligned with ground-truth patents, indicating effective control over generation granularity and mitigating the length imbalance observed in single-call and chunk-based baselines. From a content perspective, LogicTree-RAG achieves the highest coverage across all patent sections, with up to a 17.4% improvement in the Detailed Description, where fine-grained technical completeness is critical and competing methods tend to under-generate. For language quality, LogicTree-RAG consistently attains the best style scores, demonstrating strong stylistic consistency across sections.

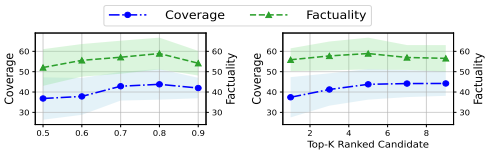
Human Expert Review. To assess practical applicability beyond automatic metrics, we conduct a blinded human expert evaluation with three patent practitioners using five criteria (i.e., Overall Quality, Legality, Claim Structure, Antecedent Basis, and Fidelity). Experts perform pairwise comparisons between our method and baselines. Figure 6(a-d) present pairwise comparisons against individual baselines, while (e-h) summarize performance across evaluation criteria. The results show that LogicTree-RAG consistently achieves strong human preference across all dimensions, indicating its practical utility in real-world scenarios. Full experimental details are provided in Appendix G.5.

4.4 Ablation and Sensitivity Analysis

Component-wise Ablation. We evaluate ablated variants of LogicTree-RAG with all other settings fixed, as shown in Table 2. Removing the logic tree or retrieval substantially shortens the outputs and degrades content quality, causing a 9.0% drop in coverage and a 13.2% degradation in factuality, respectively. This confirms the necessity of structural organization and evidence grounding. Removing semantic chunking or node-aware reranking weakens factuality and increases repetition, highlighting

the importance of coherent and non-redundant evidence selection. Moreover, removing evidence-grounded refinement σ_{R2} leads to consistent degradation across all metrics, whereas removing internal-knowledge refinement σ_{R3} yields negligible impact. Since σ_{R3} is triggered in only 8.6% of steps, it mainly serves as a fallback for under-specified cases, while the main gains come from retrieved evidence. Replacing hybrid traversal with pure DFS produces longer but less comprehensive outputs, indicating that the hybrid strategy better balances breadth and depth. Overall, the full LogicTree-RAG achieves the best performance across metrics, validating the joint contribution of structural modeling, evidence grounding, and traversal design.

Sensitivity to Key Hyperparameters. We study the effects of the semantic chunking threshold τ_s and the discriminative relevance threshold controlled by the ranked candidate k. As shown in Figure 7(a), both coverage and factuality improve as τ_s increases from low to moderate values, and peak around 0.7–0.8, suggesting that overly coarse chunks introduce noisy evidence, while overly fine-grained chunks reduce contextual sufficiency. Figure 7(b) shows that increasing k initially improves coverage by allowing more supporting evidence, but factuality peaks at a moderate value and slightly declines when weaker or redundant evidence is included. This highlights the importance of balancing evidence sufficiency and discriminative filtering, and supports the default setting of $k = 5$ as an effective compromise. Further analysis of tree growth budget parameters is provided in Appendix G.6.



(a) Semantic Chunking Threshold (b) Discriminative Relevance Threshold

Figure 7: Hyperparameter Sensitivity Analysis.

We provide additional analyses on backbone choices, agentic pipeline variants, and evidence support in Appendices G.1, G.2, and G.3, further supporting the generality and reliability of LogicTree-RAG.

4.5 Broader Implications and Limitations

This work advances controllable and evidence-grounded generative AI for long-form technical document drafting. By framing patent drafting as a representative high-stakes structured generation task, LogicTree-RAG can assist practitioners in organizing complex technical disclosures into coherent and structured drafts, potentially reducing manual effort in early-stage drafting. However, the framework is intended to assist rather than replace domain experts, and its outputs should be treated as preliminary drafts and reviewed by qualified professionals before any formal use. Beyond patents, the logic-centric paradigm extends naturally to broader long-text generation tasks that require global organization and factual grounding. While LCFO results support such generalization, further validation on diverse long-form generation settings remains an important direction for future work.

5 Related Work

Patent-related Benchmarks. Recent progress in LLMs has spurred benchmarking of patent-centric capabilities [43]. For example, IPEval [44] evaluates patent law QA, PatentEval [12] targets claims-to-abstract and next-claim generation, and HUPD-DCG [45] measures claim generation from patent descriptions. Complementary to task-specific benchmarks, Pap2Pat [16] extends evaluation to long-text patent drafting, reflecting growing interest in document-level generation. These benchmarks illustrate the evolution of patent-related evaluation from localized capabilities toward holistic long-form technical writing.

Patent-related Generation. Existing patent generation methods can be broadly grouped into three paradigms. (i) *Patent-to-patent* generates specific sections conditioned on other parts within the same patent. For instance, PGT [8] is pre-trained to support title-to-abstract and abstract-to-claim generation. PatentLMM [46] presents a large multimodal model to write descriptions of patent figures. [13] proposes a knowledge-tuning approach for initial concept discovery and content generation. (ii) *Draft-to-patent* expands patent drafts into patent-form text. PatentGPT [11] pre-trains a series of base LLMs and benchmarks them with patent generation from technical disclosure documents draft. AutoPatent [15] designs a multi-agent workflow to migrate a form of answers to technical questions to the corresponding patent draft. (iii) *Paper-to-patent* targets long-form patent drafting from research papers, exemplified by Pap2Pat [16] with outline-guided generation. However, prior methods either operate under partial-context settings or depend on externally provided (often manually crafted) drafting priors, leaving realistic long-form patent drafting with global logical organization less well addressed. See Appendix B for a comprehensive review.

6 Conclusions

Automated patent drafting exposes a fundamental bottleneck in long-form technical text generation, where sustaining globally consistent logical structure and faithful technical reasoning remains challenging. In this paper, we propose LogicTree-RAG, a logic tree-guided retrieval-augmented generation framework for long-form patent drafting that introduces a hierarchical logic tree as a global organizational backbone, where each node represents a technical element and is grounded by retrieved evidence via an evidence-guided recursive generation mechanism. To ensure precise and balanced tree construction, we develop node-aware semantic search with discriminative relevance scoring and design a hybrid traversal strategy that maps the logic tree into structurally coherent patent sections. Extensive experiments show that LogicTree-RAG consistently improves content quality and language conformity over strong long-context and multi-stage LLM-based baselines, enabling longer structured generation with high token efficiency.

Acknowledgements

We dedicate this work to the memory of Professor Beng Chin Ooi, whose wisdom, generosity, and vision have profoundly shaped our research journey. His guidance and legacy will continue to inspire us.

This research is supported by the National Natural Science Foundation of China (624B2027) and the Ministry of Education, Singapore, under an AcRF Tier 1 Grant (A-8002594-00-00).

References

[1] Edwin Mansfield. Patents and innovation: an empirical study. *Management science*, 32(2):173–181, 1986.

[2] Christopher M Kalanje. Role of intellectual property in innovation and new product development. *World Intellectual Property Organization*, 2006.

[3] Avery N Goldstein. *Patent Laws for Scientists and Engineers*. CRC Press, 2005.

[4] Charlotte Waelde, Graeme Laurie, Abbe Brown, Smita Kheria, and Jane Cornwell. *Contemporary intellectual property: law and policy*. Oxford University Press, USA, 2014.

[5] Homaira Huda Shomee, Zhu Wang, Sathya N Ravi, and Sourav Medya. A survey on patent analysis: From nlp to multimodal ai. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8545–8561, 2025.

[6] Yao Ding, Yuqing Wu, and Ziyang Ding. An automatic patent literature retrieval system based on llm-rag. *arXiv preprint arXiv:2508.14064*, 2025.

[7] Homaira Huda Shomee, Suman Kalyan Maity, and Sourav Medya. PATENTWRITER: A benchmarking study for patent drafting with llms. *CoRR*, abs/2507.22387, 2025.

[8] Dimitrios Christofidellis, Antonio Berrios Torres, Ashish Dave, Manuel Roveri, Kristin Schmidt, Sarath Swaminathan, Hans Vandierendonck, Dmitry Zubarev, and Matteo Manica. Pgt: a prompt based generative transformer for the patent domain. In *ICML 2022 workshop on knowledge retrieval and language models*, 2022.

[9] Jieh-Sheng Lee. Instructpatentgpt: training patent language models to follow instructions with human feedback. *Artificial Intelligence and Law*, pages 1–44, 2024.

[10] Juanyan Wang, Sai Krishna Reddy Mudhiganti, and Manali Sharma. Patentformer: a novel method to automate the generation of patent applications. In *Proceedings of the 2024 conference on empirical methods in natural language processing: industry track*, pages 1361–1380, 2024.

[11] Zilong Bai, Ruiji Zhang, Linqing Chen, Qijun Cai, Yuan Zhong, Cong Wang, Yan Fang, Jie Fang, Jing Sun, Weikuan Wang, et al. Patentgpt: A large language model for intellectual property. *arXiv preprint arXiv:2404.18255*, 2024.

[12] You Zuo, Kim Gerdes, Eric Villemonte de La Clergerie, and Benoît Sagot. Patenteval: Understanding errors in patent generation. In *NAACL2024-2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2024.

[13] Runtao Ren, Jian Ma, and Jianxi Luo. Large language model for patent concept generation. *Advanced Engineering Informatics*, 65:103301, 2025.

[14] Yongmin Yoo, Qiongkai Xu, and Longbing Cao. Patentscore: Multi-dimensional evaluation of llm-generated patent claims. *arXiv preprint arXiv:2505.19345*, 2025.

[15] Qiyao Wang, Shiwen Ni, Huaren Liu, Shule Lu, Guhong Chen, Xi Feng, Chi Wei, Qiang Qu, Hamid Alinejad-Rokny, Yuan Lin, et al. Autopatent: a multi-agent framework for automatic patent generation. *arXiv preprint arXiv:2412.09796*, 2024.

[16] Valentin Knappich, Anna Hätty, Simon Razniewski, and Annemarie Friedrich. PAP2PAT: benchmarking outline-guided long-text patent generation with patent-paper pairs. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 9524–9554. Association for Computational Linguistics, 2025.

[17] Xiang Liu, Peijie Dong, Xuming Hu, and Xiaowen Chu. Longgenbench: Long-context generation benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 865–883, 2024.

[18] Yuhao Wu, Ming Shan Hee, Zhiqiang Hu, and Roy Ka-Wei Lee. Longgenbench: Benchmarking long-form generation in long context llms. In *International Conference on Learning Representations*, 2025.

[19] Lekang Jiang and Stephan Goetz. Artificial intelligence exploring the patent field. *CoRR*, 2024.

[20] Fiona Murray. Innovation as co-evolution of scientific and technological networks: exploring tissue engineering. *Research policy*, 31(8-9):1389–1403, 2002.

[21] Tom Magerman, Bart Van Looy, and Xiaoyan Song. Exploring the feasibility and accuracy of latent semantic analysis based text mining techniques to detect similarity between patent documents and scientific publications. *Scientometrics*, 82(2):289–306, 2010.

[22] Joshua S Gans, Fiona E Murray, and Scott Stern. Contracting over the disclosure of scientific knowledge: Intellectual property and academic publication. *Research Policy*, 46(4):820–835, 2017.

[23] Fiona Murray and Scott Stern. Do formal intellectual property rights hinder the free flow of scientific knowledge? evidence from patent-paper pairs. *Northwestern University and MIT Sloan School: unpublished paper*, 2004.

[24] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1192–1200, 2020.

[25] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

[26] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[27] Yunpeng Huang, Jingwei Xu, Junyu Lai, Zixu Jiang, Taolue Chen, Zenan Li, Yuan Yao, Xiaoxing Ma, Lijuan Yang, Hao Chen, et al. Advancing transformer architecture in long-context large language models: A comprehensive survey. *arXiv preprint arXiv:2311.12351*, 2023.

[28] Marta R Costa-jussà, Pierre Andrews, Mariano Coria Meglioli, Joy Chen, Joe Chuang, David Dale, Christophe Ropers, Alexandre Mourachko, Eduardo Sánchez, Holger Schwenk, et al. Lcfo: Long context and long form output dataset and benchmarking. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10672–10700, 2025.

[29] Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

[30] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jian Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025.

[31] DeepSeek-AI. Deepseek-v3 technical report. *CoRR*, abs/2412.19437, 2024.

[32] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[33] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[34] Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longwriter: Unleashing 10,000+ word generation from long context llms. In *The Thirteenth International Conference on Learning Representations*, 2025.

[35] Yuhao Wu, Yushi Bai, Zhiqiang Hu, Roy Ka-Wei Lee, and Juanzi Li. Longwriter-zero: Mastering ultra-long text generation via reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2026.

[36] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

[37] Gaole He, Gianluca Demartini, and Ujwal Gadiraju. Plan-then-execute: An empirical study of user trust and team performance when using llm agents as a daily assistant. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–22, 2025.

[38] Barrett Lattimer, Patrick H Chen, Xinyuan Zhang, and Yi Yang. Fast and accurate factual inconsistency detection over long documents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1691–1703, 2023.

[39] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[40] Inez Okulska, Daria Stetsenko, Anna Kołos, Agnieszka Karlińska, Kinga Głąbińska, and Adam Nowakowski. Stylometrix: An open-source multilingual tool for representing stylometric vectors. *arXiv preprint arXiv:2309.12810*, 2023.

[41] Mauro Cettolo, Nicola Bertoldi, and Marcello Federico. The repetition rate of text as a predictor of the effectiveness of machine translation adaptation. In *Proceedings of the 11th Conference of the Association for Machine Translation in the Americas: MT Researchers Track*, pages 166–179, 2014.

[42] Wei Zhao, Michael Strube, and Steffen Eger. Discoscore: Evaluating text generation with bert and discourse coherence. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3865–3883, 2023.

[43] Lekang Jiang and Stephan M Goetz. Natural language processing in the patent domain: a survey. *Artificial Intelligence Review*, 58(7):214, 2025.

[44] Qiyao Wang, Jianguo Huang, Shule Lu, Yuan Lin, Kan Xu, Liang Yang, and Hongfei Lin. Ipeval: A bilingual intellectual property agency consultation evaluation benchmark for large language models. *CoRR*, abs/2406.12386, 2024.

[45] Lekang Jiang, Caiqi Zhang, Pascal A. Scherz, and Stefan Goetz. Can large language models generate high-quality patent claims? In *Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pages 1272–1287. Association for Computational Linguistics, 2025.

[46] Shreya Shukla, Nakul Sharma, Manish Gupta, and Anand Mishra. Patentlmm: Large multimodal model for generating descriptions for patent figures. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 20488–20496. AAAI Press, 2025.

[47] Kevin Yang, Yuandong Tian, Nanyun Peng, and Dan Klein. Re3: Generating longer stories with recursive reprompting and revision. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4393–4479, 2022.

[48] Kevin Yang, Dan Klein, Nanyun Peng, and Yuandong Tian. Doc: Improving long story coherence with detailed outline control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3378–3465, 2023.

[49] Ishita Kumar, Snigdha Viswanathan, Sushrita Yerra, Alireza Salemi, Ryan A Rossi, Franck Dernoncourt, Hanieh Deilamsalehy, Xiang Chen, Ruiyi Zhang, Shubham Agarwal, et al. Longlamp: A benchmark for personalized long-form text generation. *arXiv preprint arXiv:2407.11016*, 2024.

[50] Yiyan Xu, Jinghao Zhang, Alireza Salemi, Xinting Hu, Wenjie Wang, Fuli Feng, Hamed Zamani, Xiangnan He, and Tat-Seng Chua. Personalized generation in large model era: A survey. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24607–24649, 2025.

[51] Alireza Salemi, Cheng Li, Mingyang Zhang, Qiaozhu Mei, Weize Kong, Tao Chen, Zhuowan Li, Michael Bendersky, and Hamed Zamani. Reasoning-enhanced self-training for long-form personalized text generation. *arXiv preprint arXiv:2501.04167*, 2025.

[52] Chau Minh Pham, Yapei Chang, and Mohit Iyyer. Clipper: Compression enables long-context synthetic data generation. *arXiv preprint arXiv:2502.14854*, 2025.

[53] Chau Minh Pham, Simeng Sun, and Mohit Iyyer. Suri: Multi-constraint instruction following for long-form text generation. *URL https://arxiv. org/abs/2406.19371*, 2024.

[54] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, 2019.

[55] Barbara J Grosz, Aravind Joshi, and Scott Weinstein. Centering: A framework for modeling the local coherence of discourse. *Computational linguistics*, 21(2):203–225, 1995.

[56] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[57] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024.

[58] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[59] Qwen Team et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2(3), 2024.

Appendix

- **A. Notation Table** 15
- **B. Extended Related Work** 15
- **C. Details on Semantic Chunking** 16
- **D. Workflow of Logic Tree Construction** 16
- **E. Patent Generation Prompt** 17
- **F. Implementation Details** 19
  - *F.1. Evaluation Metrics* 19
  - *F.2. Baseline Methods* 21
  - *F.3. Hyperparameters and Decoding Configurations* 22
  - *F.4. Experimental Environment* 22
  - *F.5. Detailed Prompt Settings* 22
- **G. Supplementary Experimental Results** 25
  - *G.1. Supplementary Ablation Study* 25
  - *G.2. Supplementary Study on Agentic Planning* 25
  - *G.3. Evidence-support Analysis* 25
  - *G.4. Computational Cost Analysis* 26
  - *G.5. Human Expert Study* 26
  - *G.6. Sensitivity to Tree Growth Budget* 27
  - *G.7. Case Study* 28
- **H. Discussion on Human Evaluation and Legal Adequacy** 29

A Notation Table

To provide a comprehensive overview of the notations used throughout the paper, we present a summary of notations in Table 3 as a quick reference to facilitate the understanding and recall of each symbol.

Table 3: Notations.

Notation	Description
(X, P)	A patent-paper pair, where X is the source (pre-publication) research paper and P is the corresponding patent document.
$X = \{x_1, x_2, \ldots, x_m\}$	Source paper represented as a collection of m text spans.
$P = \{p_1, \ldots, p_n\}$	Patent document represented as a collection of n text spans.
$\tilde{X} = \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_L\}$	Semantic chunks produced from X after layout-aware semantic chunking, with $L \leq m$.
$\mathcal{T} = (V, E, r)$	Logic tree, a directed rooted tree with node set V, edge set $E \subseteq V \times V$, and root node r.
$V = \{v_1, \cdots, v_N\}$	Set of logic tree nodes, each representing a technical element (e.g., principle, step, subsystem).
v_t	Current node dequeued for recursive generation at iteration t.
v_f	Parent node of v_t in the logic tree.
V_f^{child}	Set of child nodes of v_f.
$c(v)$	Generated text fragment associated with node v.
$\mathcal{E}(v)$	Supporting evidence spans from the source paper attributed to node v, $\mathcal{E}(v) \subseteq X$.
$\mathcal{V}_{kb}$	Vectorized document knowledge base built from semantic chunks for retrieval.
$\mathcal{E}_p(\cdot)$	Paragraph-level embedding function used for semantic similarity and indexing.
$\mathcal{M}_{LLM}$	Backbone large language model.
σ_I	Instruction prompt for generating an initial draft technical description.
$\mathcal{D}_d$	Draft technical description generated by $\mathcal{M}_{LLM}(\sigma_I, \tilde{X})$ to initialize the logic tree root.
$E(\cdot)$	Expansion procedure.
$R(\cdot)$	Refinement procedure.
σ_E	Expansion instruction prompt used to generate new child nodes from retrieved context.
σ_R	Refinement instruction set; in the draft $\sigma_R = \{\sigma_{R1}, \sigma_{R2}, \sigma_{R3}\}$.
$\mathcal{Q}$	FIFO queue used for breadth-first expansion over leaf nodes in recursive generation.
$\mathcal{D}_{ctx}$	Context evidence set used to expand or refine the current node v_t.
$S_{dr}(d)$	Discriminative relevance score for candidate span d.
τ_f	Threshold for retaining a candidate span via discriminative relevance.
τ_s	Semantic chunking threshold.
$t_{\max}$	Upper bound on accumulated token length for a chunk during semantic chunking.
T_d	A subtree allocated to the Detailed Description section.

B Extended Related Work

In this section, we provide an extended review of related work, encompassing both general long-text generation methods and patent-specific drafting paradigms.

B.1 Long-text Generation

Recent advances in long-text generation have explored structured prompting, hierarchical decomposition, and personalization to improve coherence over extended outputs [35]. Early approaches, such as recursive and outline-based generation [47, 48], introduce hierarchical planning to maintain narrative consistency across long contexts. More recent work further incorporates personalization [49, 50] and reasoning-oriented training [51], enabling models to better align outputs with user intent.

With growing demand for ultra-long generation (e.g., beyond 2k tokens), new datasets and training paradigms have emerged. For instance, large-scale instruction-following datasets constructed via back-translation support long-context learning but often rely on synthetic supervision and remain limited in output length [52, 53]. Other approaches, such as LongWriter [34], leverage supervised and preference-based optimization on agent-generated long-form data to extend output length, but may inherit biases from teacher models and lack explicit control over global structure. LongWriter-Zero [35] further explores reinforcement learning from scratch to induce ultra-long generation without relying on synthetic supervision, improving length control and structural consistency. Overall, despite notable progress, existing approaches lack explicit global structure modeling and fine-grained evidence grounding, limiting their ability to ensure coherent, factual, and controllable generation over ultra-long contexts.

B.2 Patent Drafting

While prior work has studied long-form generation in general domains, patent drafting introduces additional challenges due to strict structural, legal, and evidence-grounding requirements. We next review patent drafting, covering both benchmark development and generation paradigms.

Patent-related Benchmarks. With the advancements of LLMs, the effort of testing their abilities in patent-related tasks has emerged. Recent progress in LLMs has spurred benchmarking of patent-centric capabilities [43]. For example, IPEval [44] tests LLMs for patent law QA. PatentEval [12] focuses on the tasks of claims-to-abstract and next-claim generations. HUPD-DCG [45] measures patent claim generation performance from patent descriptions. Complementary to task-specific benchmarks, Pap2Pat [16] extends evaluation to long-text patent drafting, reflecting growing interest in document-level generation. These benchmarks illustrate the evolution of patent-related evaluation from localized capabilities toward holistic long-form technical writing.

Patent-related Generation. Existing patent generation methods can be broadly grouped into three paradigms. First is *patent-to-patent*, where a section of the patent content is generated from those of specific parts in the same patent. For instance, PGT [8] is pre-trained to support multiple patent-related tasks, e.g., title-to-abstract, abstract-to-claim. PatentLMM [46] presents a large multimodal model to write descriptions of patent figures. [13] proposes a knowledge-tuning approach for initial concept discovery and content generation. Second is *draft-to-patent*, where a draft of a patent is expanded and aligned with patent format and wording. PatentGPT [11] pre-trains a series of base LLMs and benchmarks them with patent generation from technical disclosure documents draft. AutoPatent [15] generates patents from draft information, where the draft is constructed through multiple rounds of iterative question–answering with an LLM under the guidance of a reference patent document. Third is *paper-to-patent*, where an end-to-end process is studied from the original research paper to the patent draft. For example, Pap2Pat [16] leverages ground-truth patent-derived outlines to guide the full-text patent migration. However, prior methods either operate under partial-context settings or depend on externally provided (often manually crafted) drafting priors, leaving realistic long-form patent drafting with global logical organization less well addressed.

C Details on Semantic Chunking

To convert the raw content of the input document into knowledge units suitable for retrieval-augmented generation, we first perform semantic structure reconstruction based on page-level metadata extracted by PyMuPDF [2], including textual blocks, font styles, coordinates, line spans, and paragraph layouts. Using these layout cues, the system identifies the hierarchical boundaries of sections and paragraphs, thereby reconstructing the document's logical structure.

Building upon this structure, we conduct semantic chunking to produce contextually coherent text segments that serve as the foundational elements of the vectorized document knowledge base. Unlike fixed-length segmentation methods [27, 16], our approach dynamically determines chunk boundaries according to the semantic coherence and contextual dependency between adjacent paragraphs. Formally, given a document represented as an ordered sequence of paragraphs $\tilde{X} = \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_m\}$, we compute the pairwise semantic similarity $s_{i,i+1} = \text{sim}(\mathcal{E}_p(\tilde{x}_i), \mathcal{E}_p(\tilde{x}_{i+1}))$, where $i = 1, \cdots, m-1$, and $\mathcal{E}_p(\cdot)$ denotes the paragraph-level embedding function (e.g., Sentence-BERT [54]). The pairwise semantic similarity $\text{sim}(\cdot, \cdot)$ can be instantiated with different similarity metrics, while cosine similarity is adopted in this work due to its efficiency and empirical stability. Paragraphs are iteratively merged when $s_{i,i+1} \geqslant \tau_s$ and the accumulated token length does not exceed the upper bound t_{max}. This procedure yields a set of semantically coherent chunks $X = \{x_1, x_2, \ldots, x_L\}$, where $L \leq m$ represents the total number of chunks for the input document and x_i represents the i-th chunk aligned with the document's logical hierarchy. The resulting chunks constitute the minimal retrieval units for RAG, which are embedded into a high-dimensional semantic space and stored in the vector database as a vectorized document knowledge base, supporting structured querying, traceable grounding, and multi-turn semantic reasoning.

D Workflow of Logic Tree Construction

In this section, we provide the detailed algorithmic specification of the logic tree construction process used in LogicTree-RAG. As described in Section 3.5, logic tree construction constitutes the first stage of the overall patent generation pipeline, aiming to transform an unstructured research paper into a structured, evidence-grounded hierarchical representation. The procedure integrates semantic chunking, document-level vectorization, and evidence-guided recursive generation to progressively construct a logic tree that captures both the global structure and fine-grained technical details of the source document. Algorithm 2 presents the complete logic tree construction procedure.

[2] `https://pymupdf.readthedocs.io/`

Algorithm 2 Logic Tree Construction

Input: Pre-publication research paper, the backbone model $\mathcal{M}_{LLM}$, instructions $\sigma_I, \sigma_E, \sigma_R$, token length bound t_{max}, semantic chunking threshold τ_s, discriminative relevance threshold τ_f, content length budget B.

Output: Logic tree $\mathcal{T}$.

Semantic Chunking&logic tree initialization

Obtain an ordered sequence of paragraphs $X = \{x_1, x_2, \ldots, x_m\}$ via layout-aware document analysis of the input paper
for $i = 1$ **to** $m - 1$ **do**
  Compute the pairwise semantic similarity $s_{i,i+1}$ of x_i and x_{i+1} in X using Eq. 2
  if $s_{i,i+1} \geq \tau_s$ **and** $|x_i| + |x_{i+1}| \leq t_{max}$ **then**
    Merge x_i and x_{i+1} into a single segment
  else
    Retain x_i and x_{i+1} as separate segments
  end if
end for
Construct the final chunk set $\tilde{X} = \{\tilde{x}_1, \ldots, \tilde{x}_L\},\ L \leq m$
Build vectorized document knowledge base $\mathcal{V}_{kb} = \{\mathcal{E}_p(\tilde{x}_i)\}_{i=1}^{L}$
Generate a draft technical description $\mathcal{D}_d = \mathcal{M}_{LLM}(\sigma_I, \tilde{X})$
Create root node r of $\mathcal{T}$ with $c(r) = \mathcal{D}_d$ and $\mathcal{E}(r) = \tilde{X}$
Initialize the child nodes of r by prompting $\mathcal{M}_{\text{LLM}}$ to decompose the key technical components of $\mathcal{D}_d$

Evidence-Guided Recursive Generation with Node-Aware Semantic Search

Initialize FIFO queue $\mathcal{Q}$ with all leaf nodes of $\mathcal{T}$
repeat
  Dequeue front node v_t from $\mathcal{Q}$ with parent node v_f
  Obtain candidate context spans $\tilde{\mathcal{D}}_{ctx}$ using Eq. 5
  Calculate discriminative relevance score $S_{dr}(d)$ for each $d \in \tilde{\mathcal{D}}_{ctx}$ using Eq. 6
  Obtain context evidence $\mathcal{D}_{ctx} = \{d \mid S_{dr}(d) \geq \tau_f\}$
  if $\mathcal{D}_{ctx}$ is covered by $v \in \mathcal{T} \setminus V_f^{child}$ **then**
    Skip node v_t
  else
    Expand node v_t using $\mathcal{M}_{LLM}$ with prompt σ_E via Eq. 3
    Update $\mathcal{T}$ and $\mathcal{Q}$
    if Under-specified term is detected from $c(v_t)$ using $\mathcal{M}_{LLM}$ with prompt $\sigma_{R1} \in \sigma_R$ **then**
      Refine $c(v_t)$ using $\mathcal{M}_{LLM}$ with prompt in σ_R via Eq. 4
    end if
  end if
until $\mathcal{Q}$ is empty **or** $|\mathcal{T}| \geq B$
Return Logic tree $\mathcal{T}$

E Patent Generation Prompt

LogicTree-RAG employs a structured set of prompt instructions to control different stages of logic tree construction and recursive patent generation. Specifically, we design three categories of prompts: an initialization instruction σ_I, an expansion instruction σ_E, and a refinement instruction σ_R. These prompts act as a lightweight control interface for the backbone LLM, enabling evidence-grounded, controllable long-form patent drafting.

The initialization instruction σ_I is used to generate a draft technical description $\mathcal{D}_d$, which serves as the root node of the logic tree. Given a set of semantically coherent chunks extracted exclusively from the introduction of the source paper, σ_I prompts the model to synthesize a high-level technical description that captures the problem setting, core solution, and key system or method components, as detailed below.

```
### ROLE
You are an expert patent drafter and technical editor.

### TASK DESCRIPTION
Your task is to write a draft technical description of the invention, based ONLY on the provided
     document chunks of a research paper.

### INPUT
Input: a set of semantically coherent chunks derived exclusively from the paper introduction.

Document chunks:
"""
{CHUNKS}
"""

```

```
### REQUIREMENTS
Write a draft technical description that satisfies ALL requirements:
1. Capture the invention at a system and method level: clearly state (i) the technical problem, (ii) the
      core technical solution, and (iii) the key technical components.
2. Make the description expansion-ready: include explicit technical handles such as module names/roles,
     input-output signals, key variables/parameters, and the end-to-end workflow, so that each part can
     be expanded later into detailed patent sections.
3. Do not cite the paper, authors, or section numbers. Do not mention "chunks" or "retrieval".
4. Use ONLY information supported by the chunks.
5. No numbering, no bullet points, no headings.
6. Avoid vague hedges such as "may", "could", or "for example". Prefer definitive, technical phrasing.

Output ONLY the draft technical description text, with no extra commentary.
```

Listing 1: Refinement instruction σ_{R1} for under-specified term detection.

The expansion instruction σ_E governs logic tree growth. Conditioned on a node contribution and its associated context evidence, σ_E prompts the model to elaborate the contribution into finer-grained algorithmic details. The expansion instruction σ_E is specified as follows.

```
### ROLE
You are an expert in polishing patent applications, with professional knowledge of patent drafting and
     formatting standards.

### TASK DESCRIPTION
Your task is: given a specific claim contribution, extract the concrete algorithmic steps from the
     provided context and use them to expand the contribution, further specifying its algorithmic
     details.

### REQUIREMENTS
1. The output should consist of several paragraphs without numbering. The first paragraph must restate
     the given contribution point "{text}", and the subsequent paragraphs should describe the detailed
     algorithmic steps. Separate each paragraph with a blank line.
2. The expansion must remain strictly focused on the content and concept described by the given
     contribution point, without introducing broader or unrelated concepts.
3. The total output length should be longer than the input.
4. When necessary, include key formulas and definitions. Do not insert line breaks inside formulas.
5. Maintain precision and conciseness. Avoid terms such as "for example" or "may".
6. Each paragraph must be sufficiently detailed. If a step involves multiple sub-steps, describe them
     within a single paragraph using multiple sentences. The granularity of the steps should be
     consistent, with each step being necessary and indispensable.

The original contribution point is: {TEXT}\n\n
The provided context is: {CONTEXT}\n\n
```

Listing 2: Expansion instruction σ_E for LogicTree-RAG in long-form patent drafting.

The refinement instruction set $\sigma_R = \{\sigma_{R1}, \sigma_{R2}, \sigma_{R3}\}$ addresses under-specified content during recursive generation. Specifically, σ_{R1} detects under-specified terms that lack sufficient technical detail, σ_{R2} refines such terms using retrieved supporting evidence when available, and σ_{R3} completes refinement using the model's internal knowledge when no relevant evidence exists. The refinement instructions σ_R are instantiated as follows.

```
### ROLE
You are an expert patent drafter and technical editor.

### TASK DESCRIPTION
Your task is to identify under-specified terms in a patent specification node and propose how they
     should be refined.

### DEFINITION
Definition (under-specified term):
A term/phrase is under-specified if, for a technical disclosure, it lacks at least one of the following:
      a clear definition, a concrete algorithmic procedure, required inputs/outputs, parameters/
     hyperparameters, constraints/assumptions, or implementation-level details.

### INPUT
Input node content:
"""
{NODE_CONTENT}
"""

### REQUIREMENTS
1. Each "term" must be an exact substring of the input.
2. Do not rewrite the node content. Do not add any extra keys.
3. Prefer terms that are essential for technical completeness (e.g., key components, variables,
     algorithms, objectives, constraints).
```

```
4. Output ONLY the terms, one per line, with no numbering, no bullets, no extra text.
```

Listing 3: Refinement instruction σ_{R1} for under-specified term detection.

```
### ROLE
You are an expert patent drafter and technical editor. You will refine a specific under-specified term
     in a patent-specification node using the provided evidence.

### TASK DESCRIPTION
Your task is to produce an updated version of the node content that makes the target term "{TERM}" fully
      specified in a patent-appropriate and technically reproducible way.

### INPUT
Target term to refine:
"{TERM}"

Original node content:
"""
{NODE_CONTENT}
"""

Retrieved evidence (context spans):
"""
{EVIDENCE_SPANS}
"""

### REQUIREMENTS
1. All newly added technical details MUST be supported by the evidence.
2. Preserve the original wording and structure as much as possible. Only add or revise the minimal
     necessary text to specify "{TERM}".
3. When applicable, specify inputs/outputs, key variables/parameters, computation steps, constraints/
     assumptions, and any formulas explicitly stated or implied by the evidence.
4. Use formal, precise language. Avoid hedges such as "may", "could", "for example". Avoid
     conversational tone.

Refined node content:
```

Listing 4: Refinement instruction σ_{R2} for evidence-grounded targeted refinement.

```
### ROLE
You are an expert patent drafter and technical editor. You will refine an under-specified term in a
     patent-specification node.

### TASK DESCRIPTION
Your task is to update the node content to make "{TERM}" sufficiently specified for a patent
     specification, using general and widely applicable technical descriptions.

### INPUT
Target term to refine:
"{TERM}"

Original node content:
"""
{NODE_CONTENT}
"""

### REQUIREMENTS
1. Preserve the original wording and structure as much as possible. Only add or revise the minimal
     necessary text to specify "{TERM}".
2. Provide a standard, implementation-oriented specification that would be reasonable across typical
     systems (e.g., define inputs/outputs, typical parameterization, and a generic procedure).
3. Use formal, precise language. Avoid hedges such as "may", "could", "for example". Avoid
     conversational tone.

Refined node content:
```

Listing 5: Refinement instruction σ_{R3} for knowledge completion

F Implementation Details

F.1 Evaluation Metrics

We evaluate patent generation performance from both content-level and language-level perspectives. Given the length and specialized nature of patent documents, automatic evaluation requires disentangling semantic content

quality from linguistic realization. We therefore adopt established metrics that are well-suited for long-form technical text generation.

Content-level Metrics. Content-level metrics evaluate whether the generated patent conveys accurate and sufficient technical information grounded in the reference materials. All content-level metrics are computed at the document level following the evaluation protocol [16].

- **Coverage.** Coverage measures the extent to which the generated patent covers the semantic content of the reference patent. We adopt the NLI-based SCALE metric [38], which estimates entailment between two long documents by aggregating sentence-level entailment scores.

  Given a generated patent X and a reference patent R, we treat X as the *premise document* and R as the *hypothesis document*. Due to the quadratic cost of exhaustive sentence-level comparisons, SCALE samples a set of hypothesis sentences $\{h_j\}_{j=1}^{N}$ from R. For each hypothesis sentence h_j, the top-K most relevant premise chunks $\{p_{j,k}\}_{k=1}^{K}$ are retrieved from X using BM25. An NLI model then computes entailment probabilities $P_{\text{entail}}(p_{j,k}, h_j)$. The sentence-level entailment score is defined as:

$$s(h_j) = \max_k P_{\text{entail}}(p_{j,k}, h_j). \tag{7}$$

  The coverage score is computed by averaging over all sampled hypothesis sentences:

$$\text{Coverage}(X, R) = \frac{1}{N} \sum_{j=1}^{N} s(h_j). \tag{8}$$

- **Patent-grounded Factuality.** Patent-grounded factuality ($\mathcal{F}_{Pat}$) measures the extent to which the generated patent is supported by the reference patent alone. In this setting, the reference patent R serves as the premise document and the generated patent X serves as the hypothesis document:

$$\mathcal{F}_{Pat}(X, R) = \text{SCALE}(R \Rightarrow G). \tag{9}$$

- **Source-grounded Factuality.** Source-grounded factuality ($\mathcal{F}_{Src}$) measures semantic consistency when both the reference patent and the source paper are used as grounding evidence. Specifically, the premise document is formed by concatenating the reference patent R and the source paper P, while the generated patent X remains the hypothesis document:

$$\mathcal{F}_{Src}(X, R, P) = \text{SCALE}(R \cup P \Rightarrow G). \tag{10}$$

  This variant accounts for technically correct information present in the source paper but absent from the granted patent.

- **ROUGE-L.** ROUGE-L evaluates the longest common subsequence (LCS) between predicted and ground truth answers, reflecting recall-oriented similarity. It captures global structural overlap and complements BLEU's precision-based evaluation.

$$\text{ROUGE-L} = \frac{(1 + \beta^2) \cdot R_{lcs} \cdot P_{lcs}}{R_{lcs} + \beta^2 \cdot P_{lcs}}, \tag{11}$$

  where $R_{lcs} = \frac{\text{LCS}(X,Y)}{|Y|}$, $P_{lcs} = \frac{\text{LCS}(X,Y)}{|X|}$. Y is the ground truth patent, i.e., the reference patent, X is the predicted text, i.e., the generated patent. $\text{LCS}(X, Y)$ is the length of the longest common subsequence. β controls the weighting of recall versus precision.

Language-level Metrics. Language-level metrics evaluate whether the generated patent adheres to stylistic conventions of patent writing, maintains discourse coherence, and avoids degenerate generation behaviors.

- **Style.** We measure stylistic similarity using corpus-wide n-gram profiles and StyloMetrix [40], as in [16]. For n-gram profiles, we construct frequency distributions over the 1,000 most frequent n-grams for $n \in \{1, 2, 3, 4\}$ from the reference patents and the generated patents. Let $P^n_{\text{ref}}(g)$ and $P^n_{\text{gen}}(g)$ denote the normalized frequencies of n-gram g. The similarity score for each n is computed as:

$$\text{sim}_n = \sum_{g \in P^n_{\text{ref}} \cup P^n_{\text{gen}}} \left(1 - \left(\frac{P^n_{\text{ref}}(g) - P^n_{\text{gen}}(g)}{P^n_{\text{ref}}(g) + P^n_{\text{gen}}(g)}\right)^2\right). \tag{12}$$

  We additionally compute a StyloMetrix similarity score based on 196 rule-based linguistic features, including verb tense, modality, POS distributions, lexical patterns, and syntactic constructions. The final style score is computed as:

$$\text{Style} = \frac{1}{5}\left(\sum_{n=1}^{4} \text{sim}_n + \text{sim}_{\text{stylo}}\right). \tag{13}$$

- **Repetition Rate.** Repetition Rate ($\mathcal{R}_r$) is defined as the fraction of n-grams that appear more than once within a text [41], given by:

$$\mathcal{R}_r = \frac{\#\{\text{n-grams occurring more than once}\}}{\#\{\text{total n-grams}\}}. \tag{14}$$

  To capture long-range repetition and degeneration effects, $\mathcal{R}_r$ is computed over sliding windows of 256 tokens and averaged across windows.

- **Coherence.** Discourse-level coherence is evaluated using DiscoScore [42], a coherence metric based on Centering Theory [55] and contextualized representations from BERT [56]. DiscoScore assesses the consistency of entity transitions and discourse structure across adjacent sentences, producing a document-level coherence score. Given a hypothesis document X, let $\mathbf{S}_{\text{gen}} \in \mathbb{R}^{n \times d}$ denote the matrix of sentence embeddings, where n is the number of sentences and d is the embedding dimension. A sentence graph $\mathcal{G}_{\text{gen}} = (V, A_{\text{gen}})$ is constructed, where each node corresponds to a sentence and the adjacency matrix A_{gen} encodes discourse connectivity between sentence pairs. Sentence embeddings are updated via graph propagation $\hat{\mathbf{S}}_{\text{gen}} = (A_{\text{gen}} + I)\mathbf{S}_{\text{gen}}$, where I is the identity matrix.

  The same procedure is applied to the reference document R to obtain $\hat{\mathbf{S}}_{\text{ref}}$. A document-level representation is then computed by aggregating sentence embeddings, yielding $\mathbf{G}_{\text{gen}}$ and $\mathbf{G}_{\text{ref}}$. The final DiscoScore is defined as the cosine similarity between the two graph-level representations:

$$\text{DiscoScore}(X, R) = \cos(\mathbf{G}_{\text{gen}}, \mathbf{G}_{\text{ref}}). \tag{15}$$

Together, these metrics offer a comprehensive evaluation of patent generation quality that captures content accuracy, linguistic conventions, discourse coherence, and long-form generation behavior.

F.2 Baseline Methods

For heuristic baselines, including the ground-truth patent and source paper, no additional model inference is performed, and the documents are directly treated as generated outputs for metric computation.

Single LLM-call baselines generate the entire patent document in a single LLM invocation by conditioning on the full input paper using long-context prompting. All baselines support 128k-token contexts, enabling full-paper ingestion and long-text drafting. No intermediate structure or iterative refinement is employed. Specifically, Kimi K2 [29] adopts a Mixture-of-Experts (MoE) architecture with 1T total parameters and 32B activated per token, and is pre-trained on 15.5T tokens with carefully engineered optimization strategies to ensure stable training at scale. Qwen3-80B [30] follows the Qwen3-Next design and integrates hybrid attention mechanisms and high-sparsity MoE layers, enabling efficient context modeling under ultra-long context lengths while maintaining computational efficiency. Qwen3-Max [30] is a proprietary flagship model exceeding one trillion parameters, pre-trained on 36T tokens, and inherits the MoE-based design of the Qwen3 series with enhanced load balancing and stability-oriented training strategies. DeepseekV3 [31] is an MoE model with 671B total parameters and 37B activated per token, combining Multi-head Latent Attention and DeepSeekMoE architectures, and is pre-trained on 14.8T tokens with multi-token prediction objectives to support efficient and stable large-scale training. GPT-5 [32] and Gemini-2.5 Pro [33] are proprietary frontier LLMs evaluated under deep thinking mode, which enables stronger deliberative reasoning and more careful long-context synthesis. They serve as strong closed-source baselines for assessing whether advanced reasoning-oriented prompting alone can produce coherent and complete long-form patent drafts without explicit structural orchestration. Collectively, these models provide strong capacity for long-context representation and serve as competitive backbone LLMs for single LLM-call patent generation. Unless otherwise specified, these baselines use the same decoding configurations as LogicTree-RAG.

Furthermore, we include Chunk-based Outline-guided Patent Generation (COPGEN) [16] and its supervised fine-tuned (SFT) variant as representative multi-stage LLM-based patent generation methods. Specifically, COPGEN chunks the outline, selects the most relevant parts of the source paper depending on the chunks' outline bullet points, and prompts an LLM to generate the patent text for that chunk. To ensure a fair comparison under our problem setting, where no ground-truth patent outline is available at inference time, we report two variants of COPGEN. The first follows the outline-empty configuration, where only section titles are provided. For completeness, we also report results obtained using the long-version outlines provided by the Pap2Pat benchmark. For the SFT variant, only the backbone LLM is fine-tuned on the Pap2Pat training set (1,000 paper-patent pairs), and no test data is used during training.

In addition, we consider recent long-form generation frameworks LongWriter [34] and LongWriter-Zero [35], which decompose long-form generation into multi-stage processes to improve global coherence and content planning. We also include general-purpose agentic paradigms, including ReAct [36] and Plan-then-Execute [37], which enable LLMs to perform step-wise planning and tool use without relying on hard-coded structural templates. To adapt them to our setting, we expose our core components, i.e., semantic chunking, node-aware retrieval, and evidence-guided generation, as callable tools, and replace the logic-tree-based orchestration with generic agent planning strategies.

F.3 Hyperparameters and Decoding Configurations

In this subsection, we summarize the hyperparameters and decoding configurations adopted for LogicTree-RAG and the baseline methods. LogicTree-RAG adopts Qwen3-80B [30] as the backbone LLM throughout all experiments, unless stated otherwise. We set the semantic chunking threshold τ_s to 0.8, and the accumulated token length t_{max} to 512. For node-aware semantic search, we first retrieve up to 100 candidate evidence spans $\tilde{\mathcal{D}}_{ctx}$ from the document knowledge base $\mathcal{V}_{kb}$. These candidates are then re-ranked using the discriminative relevance score. Rather than using a fixed score threshold, we retain the top-5 candidates and set the discriminative relevance threshold τ_f adaptively as the score of the 5-th ranked candidate. To ensure controllable expansion during evidence-guided recursive generation, we impose a content length budget B that limits the maximum size of the logic tree. Specifically, the recursive process terminates when the number of nodes in the tree reaches 64, preventing unbounded expansion while preserving sufficient structural coverage. In addition, we constrain the maximum depth of the logic tree to 3 (with the root node at depth 0), which avoids overly deep recursive expansion and encourages a balanced hierarchical structure. Unless otherwise specified, these values are fixed across all experiments.

Reported results in Section 4.2 are evaluated on the Pap2Pat test set, which consists of 500 paper-patent pairs. For the SFT variant of COPGEN, fine-tuning is conducted exclusively on the Pap2Pat training set containing 1,000 pairs. All LLM-based methods use identical decoding configurations, with a temperature of 0.3, top-p of 1.0, and a maximum generation length of 8,192 tokens. The SFT variant of COPGEN fine-tunes the backbone LLM Llama-3 8B [57] using LoRA [58]. Following [16], we adopt a maximum sequence length of 8,192 tokens and train the model for 3 epochs with a batch size of 32. The learning rate is set to 3.16×10^{-4} with a cosine scheduler and a warmup ratio of 0.1. For LoRA-specific configurations, we use a rank of 128, a scaling factor (α) of 60, and a dropout rate of 0.05.

F.4 Experimental Environment

All experiments are conducted on a server equipped with an AMD EPYC 9334 CPU @ 2.7 GHz (32 cores), 128 GB of memory, and an NVIDIA H100 NVL with CUDA 12.9. The operating system is Ubuntu 24.04 running Linux kernel 6.8.0-79-generic. LLMs used in our experiments are accessed via standardized cloud-based inference APIs provided by Alibaba Cloud Bailian.

F.5 Detailed Prompt Settings

This subsection details the prompt configurations used in our experiments. For COPGEN and its variants, we adopt the prompt settings from the publicly released implementation[3]. For single LLM-call baselines, we employ a unified prompt template following COPGEN to ensure a fair and consistent comparison, as described below.

```
### ROLE
You are a highly skilled patent attorney with decades of experience in drafting high-quality patent applications.
You assist scientists in transforming their scientific discoveries into lucrative patents.

### TASK DESCRIPTION
Your task is to draft a patent application.

### INPUTS
As input, you will be provided a research paper and a patent outline, each serving a distinct purpose.

1. Research Paper:
The research paper describes a novel invention to be patented. Your task is to extract the invention from the paper and write a patent application for it.

2. Patent Outline:
The patent outline summarizes the desired discourse structure of the patent document. Use this outline as a rough guidance during drafting.

### GUIDELINES
* Copy the headings from the outline exactly. You must include only the headings provided in the outline.

* You must always write complete sentences and avoid keywords, bullet lists and enumerations.
* The patent must act as a standalone document, therefore do not refer to the research paper in the patent.
```

Listing 6: System prompt for single LLM-call baselines in long-form patent drafting.

[3] https://github.com/boschresearch/Pap2Pat

````
Here is the outline of the desired patent application.
Per bullet point, write roughly 4000 words.

### DESCRIPTION OF THE INVENTION
In the example above, each line beginning with '#' is a bullet point.
```md
# DESCRIPTION

## FIELD OF THE INVENTION

## BACKGROUND OF THE INVENTION

## SUMMARY OF THE INVENTION

## DETAILED DESCRIPTION
```
You need to draft a complete patent application that strictly follows the outline's section order and headings.
Do not skip any bullet points. Use formal patent language. Remember what you have already written and do not repeat yourself.

### RESEARCH PAPER
Here is the research paper that describes the invention:
```
# Introduction

Safe delivery of a highly conformal dose distribution to a well-defined target volume in radiotherapy is not an easy task. It has become increasingly challenging due to the advent of advanced treatment techniques such as intensity-modulated radiation therapy (IMRT) and volumetric-modulated arc therapy (VMAT). Though a downward trend in radiotherapy incident rates has been indicated by several reports (1, 2), severe incidents with detrimental effects, including death, have been reported recently and received public attention. (3)  Radiotherapy is a complicated, multistep, multiperson process, and errors can occur at any point. One of the prominent causes for radiotherapy incidents is geometric miss due to incorrect patient setup, which leads to the treatment of incorrect body parts with more than 1 cm spatial discrepancy (3, 4). Geometric miss can result in significant underdose to the target, which can cause tumor recurrence, and overdose to healthy tissue with severe normal tissue complications. In hypofractionated radiotherapy, such as stereotactic body radiotherapy (SBRT), it may result in even more severe morbidity than traditional radiotherapy.

Several reports have identified geometric miss caused by incorrect patient setup as the leading cause of radiotherapy errors. In a study of 100 radiotherapy incidents reported internationally, (3) the frequency of incidents due to incorrect patient setup was reported to be 21%. Excluding the 44 incidents involving brachytherapy, in which patient setup accuracy was less of an issue, this frequency would rise significantly to 37.5%. (3)  Clark et al. (2)  analyzed clinical incidents reported internally within a large academic center between 2007 and 2009, and reported that 14 out of 41 critical, major, or serious incidents were geographic misses. Among the 14 incidents, one was caused by wrong target identification in treatment planning, while the other 13 incidents were due to shifting errors at patient setup. In an online report analyzing event causes for 230 misadministrations reported between 2001 and 2009 in New York from the New York State Department of Health, incorrect body part treated was the leading type of errors at 46%, most often due to incorrect patient setup. (5)

The challenge in patient setup is to accurately localize the patient to the same treatment position as planned in each treatment session. Over the past decade, there has been a rapid expansion in technological tools to facilitate accurate patient localization. Examples include immobilization devices with couch indexing capability and image-guided radiotherapy (IGRT). Indexed immobilization devices not only ensure repeatable patient fixation but also reduce patient setup errors by providing initial approximate target localization. X-ray-based image-guided technologies, such as cone-beam CT (CBCT), enable visualization of internal anatomy with sufficient soft tissue contrast and allow corrections for misalignment or interfraction motion through registration with reference CT images (6). IGRT systems capable of continuous tracking, such as AlignRT (Vision RT Ltd., London, UK), C-Rad Sentinel (C-Rad AB, Uppsala, Sweden), SonArray (Varian Medical Systems, Palo Alto, CA), and ExacTrac Optical Tracking System (BrainLAB, Heimstetten, Germany), have also been developed for patient setup guidance. These systems have the potential to eliminate patient setup errors and significantly reduce setup uncertainty, as demonstrated by multiple studies. Bissonnette and Medlam (1) reported a 50 percent overall decrease in incidents caused by localization errors following the widespread introduction of IGRT on six of their 16 linear accelerators. Clark et al. (2) noted that none of the 13 geometric misses occurred on treatment units equipped with daily image guidance systems.


However, the use of advanced technological equipment does not guarantee that radiotherapy is immune to setup errors (7, 8). Patient setup is a process that cannot be fully automated and is therefore subject to human error. Several contributing factors have been identified. First, sufficient formal training is not always provided to personnel who operate the devices and interpret the results. Incorrect interpretation can lead to improper adjustment of treatment position and consequently to setup errors. For example, when treating the thoracic spine, there is a significant risk of targeting the wrong vertebral body (9). Due to similarities in bony anatomy in this region, incorrect alignment may occur as a result of misregistration using orthogonal imaging or CBCT.
````
````

```
      Second, as the complexity of treatment tools increases, the complexity of device control and the
      workload for therapists also increase, as reflected by the growing number of computer monitors in
      control rooms (8). In the absence of streamlined workflows and standardized controls, therapists
      may lose attention to the correctness of treatment delivery. Third, although most IGRT devices
      provide high geometric precision (10), they have inherent limitations that may compromise patient
      safety. For example, commonly used radiographic systems such as CBCT do not track patient position
      changes and therefore represent patient position only at the time of image acquisition. After CBCT
      imaging, the treatment couch may be moved for specific reasons, such as clearance checks, and not
      returned to the intended treatment position. Modern delivery systems include interlocks to detect
      such errors but also allow user overrides. If an interlock is inappropriately overridden, the
      patient may be treated at an incorrect site. In addition, CBCT and most radiographic-based IGRT
      systems cannot be used for noncoplanar setups, resulting in discrepancies between imaging and
      treatment positions when noncoplanar beams are applied (11). Continuous tracking systems such as
      ExacTrac Optical Tracking System, AlignRT, and C-Rad Sentinel are typically employed only for
      limited disease sites in current clinical practice. Therapists operating machines equipped with
      multiple IGRT systems often switch between devices depending on the treatment site, further
      increasing operational complexity. Fourth and most importantly, immediate and independent position
      verification is not always available to confirm that the patient has been positioned exactly as
      planned, particularly when the limitations of the employed IGRT systems are encountered (8).

In this paper, we present an efficient automatic patient safety system (PSS) that addresses factors
      contributing to gross setup errors and safeguards patient treatment. The system uses CCD cameras to
       track a single infrared reflective marker affixed to the patient's skin or immobilization device.
      It is a general-purpose system applicable to all disease sites and provides continuous and
      independent verification of patient setup accuracy. With a fully automated workflow, the PSS serves
       as an effective complement to existing IGRT systems to enhance patient safety in radiotherapy.


# Materials And Methods

## A. System Overview

The system consists of a pair of Polaris CCD cameras (Northern Digital Inc., Waterloo, Ontario) mounted
      on the treatment room ceiling (see Fig. 1(a)), a single IRRM affixed to the patient's skin or
      immobilization device, and a set of in-house developed software. The software communicates with the
       control box of the cameras through a standard serial cable and a proprietary data cable. An
      internal processing unit performs triangulation to determine the coordinates of IRRMs in its native
       coordinate system. The three-dimensional (3D) coordinates of up to 50 IRRMs can be transmitted to
      the computer simultaneously. System calibration, workflow, and daily quality assurance (QA)
      procedures are described in the following sections.

## B. System Calibration

A calibration procedure was developed to convert the camera's native coordinate system to the absolute
      room coordinate system. A calibration jig consisting of five commercial IRRMs (Fig. 1(b)) was used
      for this purpose. The coordinates of each IRRM relative to the center of the jig were determined
      via CT scan and collected in a 5x3 matrix A. Under the guidance of CBCT, the jig was placed on the
      treatment couch with its center aligned with the imaging isocenter. The congruence of the imaging
      isocenter and the machine isocenter was confirmed using a ball bearing phantom (6). The coordinates
       of the IRRMs were sampled by the cameras and collected into another matrix B. A rotation matrix R
      and a translation matrix S were estimated by minimizing the residual error between the transformed
      and reference coordinates.

This is known as the relative pose problem in computer vision and has been studied extensively. 12,  13,
        14  In this work, we employed a general analytical solution via singular value decomposition (SVD
      ), (14)  and the steps were summarized as follows: The centroids (Ac) and (Bc) were subtracted from
       A and B, respectively, to align the origins of the two coordinate systems. The SVD of BT A was
      computed as \(B^{T}A = \text{UWV}^{T}\) where U and V are unitary matrices, and W is a diagonal
      matrix whose diagonal entries are equal to the singular values of BT A.

3. The solutions of R and S were given as \(\begin{matrix} {R = U\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0
      \\ 0 & 0 & {\text{det}(UV^{T})} \end{pmatrix}V^{T}} \\ {\quad\quad\quad\quad\quad\quad\quad\quad\
      quad\quad\quad\quad\text{and},} \\ {S = A_{c} - B_{c}R} \end{matrix}\)

## System Workflow

Great emphasis was placed on developing a smooth clinical workflow for the system. The workload added to
       the therapists should be kept to a minimum, which was critical for the system to be adopted into
      clinic use. A minimal workload would also help minimize the possibilities of human errors. The
      workflow of the PSS is described in the following sections as the preparation stage and the
      treatment stage. Figure 2 shows a demonstrative workflow of the system when used on head and neck (
      H&N) patients treated with aquaplastic face masks.

...

# Conclusions

We have developed an efficient automatic general-purpose PSS to prevent gross setup errors in
      radiotherapy. The system provides real-time independent position verification for the treatment of
      all disease sites based on optical-tracking technology and is independent of treatment room,
      treatment machine, couch top design, immobilization device, or patient internal anatomy. The system
```

```
      has been well adopted for use in a busy clinical environment because of its seamless workflow and
     effectiveness in catching setup errors. Due to its advantages of continuous tracking capability, no
      radiation dose, and fully automated clinical workflow, it serves as an ideal complement to complex
      IGRT systems in ensuring patient safety in radiotherapy.
'''
```

Listing 7: User prompt for single LLM-call baselines in long-form patent drafting.

G Supplementary Experimental Results

G.1 Supplementary Ablation Study

In this subsection, we evaluate the robustness of LogicTree-RAG by instantiating the framework with different backbone LLMs, while keeping all other components and hyperparameters unchanged. As shown in Table 4, LogicTree-RAG consistently produces long and well-structured patent drafts across all backbones, indicating that controlling the tree size enables stable and sufficient output length independent of the underlying model. Content-level metrics such as coverage and factuality remain stable, while language-level metrics, including style and repetition rate, gradually improve with stronger backbones. Notably, when both LogicTree-RAG and COPGEN [16] are instantiated with the same Llama-3-8B backbone, LogicTree-RAG achieves substantially better performance and approaches the performance of the SFT-enhanced COPGEN variant, highlighting the effectiveness of the proposed logic-centric generation framework without relying on task-specific fine-tuning. Overall, these results demonstrate that the gains of LogicTree-RAG stem primarily from its structured generation paradigm rather than dependence on a particular backbone model.

Table 4: Performance of LogicTree-RAG with different backbone LLMs.

Method	Output Tokens	Content-level				Language-level			Time (s) / Patent (Token)
		Coverage ↑	Factuality ↑		Semantic Similarity ↑	Style ↑	Repetition Rate $\mathcal{R}_r$ ↓	Coherence ↑	
			$\mathcal{F}_{Pat}$	$\mathcal{F}_{Src}$					
Heuristic Skyline									
Reference Patent (GT)	18.16k	89.05	88.74	88.80	100	100	13.91	100	-
Source Paper	8.03k	44.77	46.06	88.68	39.37	39.26	8.39	98.60	-
Outline (Long)	1.43k	39.11	61.79	61.93	10.16	23.22	20.14	85.69	-
LogicTree-RAG									
Llama3-8B [57]	16.25k	40.25	56.70	61.95	35.74	55.89	7.16	96.98	209 (12.86)
Qwen2.5-72B [59]	17.82k	42.08	57.42	66.09	34.89	62.28	5.87	97.56	260 (14.58)
DeepseekV3 [31]	19.15k	43.20	58.21	64.83	37.95	58.24	5.86	97.10	320 (16.69)
Qwen3-80B [30]	20.22k	43.79	58.92	66.34	38.91	65.68	4.14	97.45	311 (15.38)

G.2 Supplementary Study on Agentic Planning

To directly evaluate the role of structured planning, we compare LogicTree-RAG with two representative agentic paradigms, ReAct [36] and Plan-then-Execute [37]. To ensure a fair comparison, we expose our core modules (i.e., semantic chunking, node-aware retrieval, and evidence-guided generation) as callable tools, and replace the logic-tree-based orchestration with generic agent planning strategies. This design isolates the effect of explicit structural planning (logic tree) versus generic agentic decomposition, while keeping underlying capabilities comparable.

As shown in Table 5, despite using the same underlying tools, the agentic variants exhibit lower coverage, weaker factual grounding, and higher repetition, indicating incomplete and less controlled generation. In contrast, LogicTree-RAG achieves more balanced and consistent outputs, with superior performance across both content-level and language-level metrics. These results suggest that generic agentic planning alone is insufficient for long-form structured generation. Instead, explicit global structure modeling via the logic tree is critical for coordinating content organization, evidence grounding, and generation control.

G.3 Evidence-support Analysis

To assess retrieval-grounded generation at the evidence level, we evaluate whether generated content is supported by retrieved evidence for each retrieval-triggered node, using an LLM-as-judge (GPT-5) and human audit. Each case is categorized as fully supported, partially supported, or unsupported. We further conduct human auditing

Table 5: Performance comparison of LogicTree-RAG against agentic baselines for patent drafting.

Method	Output Tokens	Content-level				Language-level			Time (s) / Patent (Token)
		Coverage ↑	Factuality ↑		Semantic Similarity ↑	Style ↑	Repetition Rate $\mathcal{R}_r$ ↓	Coherence ↑	
			$\mathcal{F}_{Pat}$	$\mathcal{F}_{Src}$					
LogicTree-RAG	**20.22k**	**43.79**	**58.92**	**66.34**	**38.91**	**65.68**	**4.14**	**97.45**	311 (15.38)
↪w/ ReAct	16.86k	40.28	55.98	64.85	36.86	65.52	10.24	96.18	336 (19.93)
↪w/ Plan-Execute	18.58k	42.02	56.35	64.96	35.91	64.97	8.95	96.59	325 (17.49)

on 20 randomly sampled test cases under the same criteria. Results are summarized in Table 6. LogicTree-RAG achieves a substantially higher proportion of fully supported content and fewer unsupported cases than COPGEN, indicating stronger attribution faithfulness and more effective use of retrieved evidence. Human auditing yields consistent trends, confirming that retrieved evidence is actively used to support generation rather than being passively retrieved. These findings align with the ablation results in Table 2, where removing evidence-grounded refinement σ_{R2} consistently degrades performance, while removing internal refinement σ_{R3} has a negligible impact. These results suggest that the gains of LogicTree-RAG primarily arise from improved evidence-grounded generation, rather than increased pipeline complexity or auxiliary mechanisms.

Table 6: Evidence-support analysis of retrieved evidence and generated content.

Method	Fully (%)	Partial (%)	Unsupported (%)
LLM-as-judge			
COPGEN	70.4	20.2	9.4
LogicTree-RAG	85.6	12.0	2.4
Human Audit			
COPGEN	60	30	10
LogicTree-RAG	75	25	0

G.4 Computational Cost Analysis

To improve transparency, we report a detailed cost breakdown of LogicTree-RAG along the workflow pipeline. As shown in Table 7, the overall cost is well controlled, with a balanced distribution across stages. In particular, recursive logic tree construction introduces only moderate overhead, while the majority of cost arises from traversal and generation. Importantly, the recursion depth is explicitly bounded by tree budget hyperparameters (see Appendix F.3), preventing uncontrolled expansion and ensuring a favorable quality-efficiency trade-off rather than gains driven by excessive API usage.

Table 7: Supplementary computational cost analysis.

Computational Cost (/patent)	Cost Breakdown
LLM Calls: 20	chunking&tree init 5%, tree construction 45%, traversal 50%
Runtime: 311s	chunking&tree init 16%, tree construction 39%, traversal 45%
Total Input Tokens: 7.60k	chunking&tree init 29%, tree construction 37%, traversal 34%

G.5 Human Expert Study

We conduct a human expert evaluation with three patent practitioners to assess the practical quality of generated patents. We randomly select 10 samples from the test set. For each sample, practitioners are provided with the source paper, the generated patent, and the reference patent, along with the corresponding outlines (long version) to ensure fair comparison. Experts perform pairwise comparisons (win/loss/tie) between LogicTree-RAG and each baseline. We adopt five evaluation criteria covering both legal correctness and technical quality, as defined in Listing 8. All evaluations are conducted in a blinded manner, where the identities of methods are anonymized.

The win-rate results are summarized in Table 8. LogicTree-RAG consistently outperforms all baselines across evaluation criteria, achieving strong human preference in overall quality and technical correctness. In particular, LogicTree-RAG shows substantial advantages in Claim Structure and Antecedent Basis, indicating improved structural coherence and consistency in long-form patent drafting. Gains in Fidelity further demonstrate that the generated patents remain closely grounded in the source content.

```
### Expert Evaluation Criteria
* Overall Quality: overall usefulness as a patent draft for practitioner revision
* Legality: conformity to basic patent drafting and legal-form requirements
* Claim Structure: clarity and correctness of claim organization and dependencies
* Antecedent Basis: consistency of terminology and antecedent basis
* Fidelity: faithfulness to the technical content of the source paper
```

Listing 8: Definitions of evaluation criteria used in human expert assessment.

Table 8: Win-rate results of LogicTree-RAG in human expert evaluation.

Method	Overall	Legality	Claim Structure	Antecedent Basis	Fidelity
Ours vs Qwen3-80B	75.7	69.8	84.6	84.8	88.5
Ours vs DeepseekV3	84.4	65.1	88.5	90.1	90.2
Ours vs LongWriter	78.8	76.5	90.6	92.0	85.9
Ours vs COPGEN	70.5	65.7	85.2	82.1	78.5

G.6 Sensitivity to Tree Growth Budget

We further analyze the sensitivity of LogicTree-RAG to the tree growth budget by varying the maximum tree depth and the maximum number of tree nodes, while keeping all other components and hyperparameters fixed. As shown in Figure 8, increasing the tree budget consistently improves coverage, indicating that deeper or broader trees enable more comprehensive exploration of the document structure. However, the gains diminish beyond moderate budgets, suggesting that additional expansions primarily capture long-tail details. In particular, the default configuration, i.e., maximum depth = 3 and maximum number of nodes = 64, lies in this favorable regime, achieving high coverage with reasonable efficiency. In contrast, token efficiency decreases as the tree budget grows, due to the increased retrieval and generation overhead incurred by additional nodes. These results reveal a clear quality–cost trade-off and demonstrate that LogicTree-RAG allows controllable scaling, where moderate tree budgets achieve a favorable balance between coverage and efficiency.

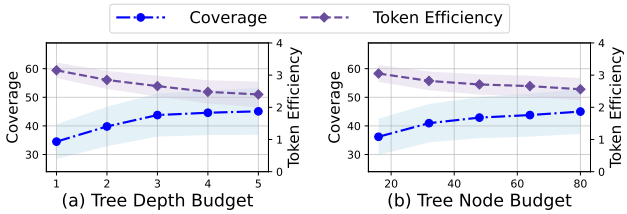


Figure 8: Sensitivity to tree growth budget in LogicTree-RAG.

G.7 Case Study

To qualitatively assess the structural fidelity of long-form patent generation, we present a case study comparing a patent generated by our method with its ground-truth counterpart, as shown in Figure 9 and 10. Both documents are visualized using section-level color coding to facilitate direct comparison of document organization and length distribution. As observed, the generated patent closely matches the reference patent in overall length and preserves a well-balanced allocation across major sections.

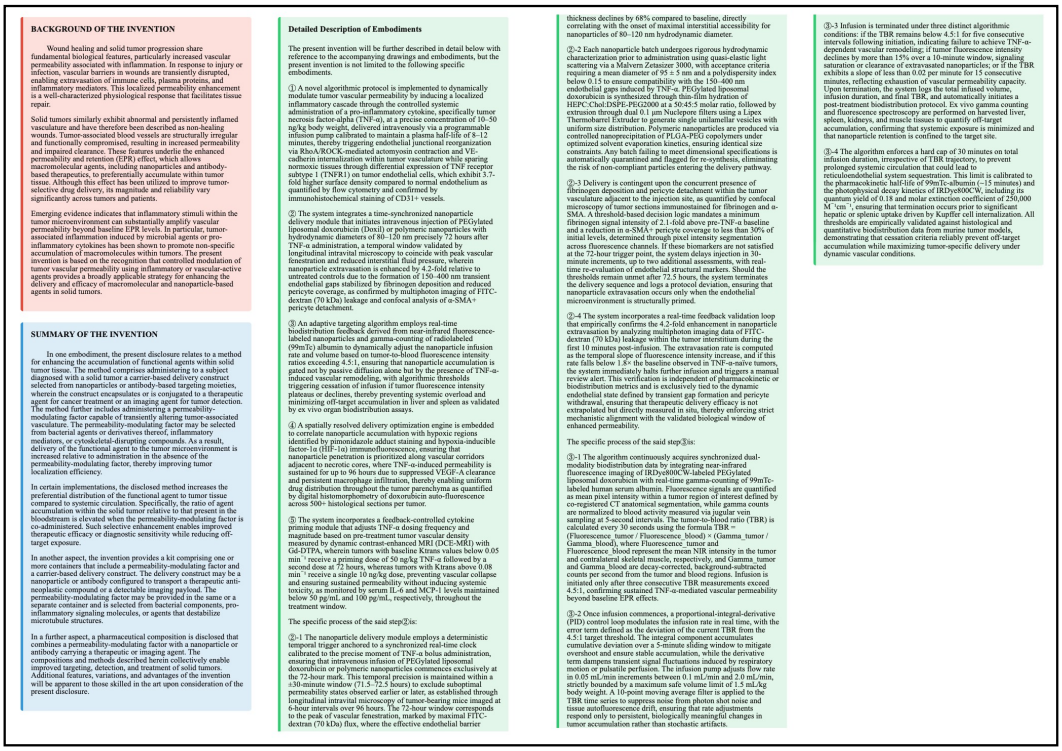
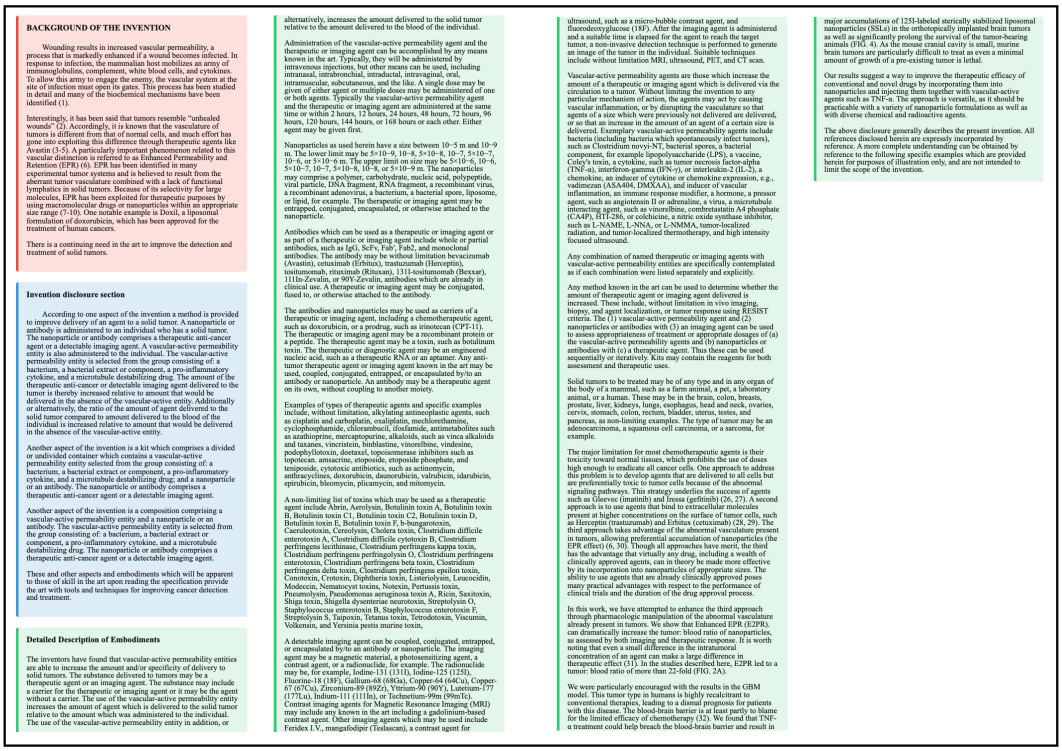


Figure 9: Qualitative case study illustrating a patent example generated by LogicTree-RAG.



BACKGROUND OF THE INVENTION

Wounding results in increased vascular permeability, a process that is markedly enhanced if a wound becomes infected. In response to infection, the mammalian host mobilizes an army of immunoglobulins, complement, white blood cells, and cytokines. To allow this army to engage the enemy, the vascular system at the site of infection must open its gates. This process has been studied in detail and many of the biochemical mechanisms have been identified (1).

Interestingly, it has been said that tumors resemble "unhealed wounds" (2). Accordingly, it is known that the vasculature of tumors is different from that of normal cells, and much effort has gone into exploiting this difference through therapeutic agents like Avastin (3-5). A particularly important phenomenon related to this vascular distinction is referred to as Enhanced Permeability and Retention (EPR) (6). EPR has been identified in many experimental tumor systems and is believed to result from the aberrant tumor vasculature combined with a lack of functional lymphatics in solid tumors. Because of its selectivity for large molecules, EPR has been exploited for therapeutic purposes by using macromolecular drugs or nanoparticles within an appropriate size range (7-10). One notable example is Doxil, a liposomal formulation of doxorubicin, which has been approved for the treatment of human cancers.

There is a continuing need in the art to improve the detection and treatment of solid tumors.

Invention disclosure section

According to one aspect of the invention a method is provided to improve delivery of an agent to a solid tumor. A nanoparticle or antibody is administered to an individual who has a solid tumor. The nanoparticle or antibody comprises a therapeutic anti-cancer agent or a detectable imaging agent. A vascular-active permeability entity is also administered to the individual. The vascular-active permeability entity is selected from the group consisting of: a bacterium, a bacterial extract or component, a pro-inflammatory cytokine, and a microtubule destabilizing drug. The amount of the therapeutic anti-cancer or detectable imaging agent delivered to the tumor is thereby increased relative to amount that would be delivered in the absence of the vascular-active entity. Additionally or alternatively, the ratio of the amount of agent delivered to the solid tumor compared to amount delivered to the blood of the individual is increased relative to amount that would be delivered in the absence of the vascular-active entity.

Another aspect of the invention is a kit which comprises a divided or undivided container which contains a vascular-active permeability entity selected from the group consisting of: a bacterium, a bacterial extract or component, a pro-inflammatory cytokine, and a microtubule destabilizing drug; and a nanoparticle or an antibody. The nanoparticle or antibody comprises a therapeutic anti-cancer agent or a detectable imaging agent.

Another aspect of the invention is a composition comprising a vascular-active permeability entity and a nanoparticle or an antibody. The vascular-active permeability entity is selected from the group consisting of: a bacterium, a bacterial extract or component, a pro-inflammatory cytokine, and a microtubule destabilizing drug. The nanoparticle or antibody comprises a therapeutic anti-cancer agent or a detectable imaging agent.

These and other aspects and embodiments which will be apparent to those of skill in the art upon reading the specification provide the art with tools and techniques for improving cancer detection and treatment.

Detailed Description of Embodiments

The inventors have found that vascular-active permeability entities are able to increase the amount and/or specificity of delivery to solid tumors. The substance delivered to tumors may be a therapeutic agent or an imaging agent. The substance may include a carrier for the therapeutic or imaging agent or it may be the agent without a carrier. The use of the vascular-active permeability entity increases the amount of agent which is delivered to the solid tumor relative to the amount which was administered to the individual. The use of the vascular-active permeability entity in addition, or alternatively, increases the amount delivered to the solid tumor relative to the amount delivered to the blood of the individual.

Administration of the vascular-active permeability agent and the therapeutic or imaging agent can be accomplished by any means known in the art. Typically, they will be administered by intravenous injections, but other means can be used, including intranasal, intrabronchial, intradactal, intravaginal, oral, intramuscular, subcutaneous, and the like. A single dose may be given of either agent or multiple doses may be administered of one or both agents. Typically the vascular-active permeability agent and the therapeutic or imaging agent are administered at the same time or within 2 hours, 12 hours, 24 hours, 48 hours, 72 hours, 96 hours, 120 hours, 144 hours, or 168 hours or each other. Either agent may be given first.

Nanoparticles as used herein have a size between 10^{-5} m and 10^{-9} m. The lower limit may be 5×10^{-9}, 10^{-8}, 5×10^{-8}, 10^{-7}, 5×10^{-7}, 10^{-6}, or 5×10^{-6} m. The upper limit on size may be 5×10^{-6}, 10^{-6}, 5×10^{-7}, 10^{-7}, 5×10^{-8}, 10^{-8}, or 5×10^{-9} m. The nanoparticles may comprise a polymer, carbohydrate, nucleic acid, polypeptide, viral particle, DNA fragment, RNA fragment, a recombinant virus, a recombinant adenovirus, a bacterium, a bacterial spore, liposome, or lipid, for example. The therapeutic or imaging agent may be entrapped, conjugated, encapsulated, or otherwise attached to the nanoparticle.

Antibodies which can be used as a therapeutic or imaging agent or as part of a therapeutic or imaging agent include whole or partial antibodies, such as IgG, $ScFv$, Fab', $Fab2$, and monoclonal antibodies. The antibody may be without limitation bevacizumab (Avastin), cetuximab (Erbitux), trastuzumab (Herceptin), iositumomab, rituximab (Rituxan), 131I-tositumomab (Bexxar), 111In-Zevalin, or 90Y-Zevalin, antibodies which are already in clinical use. A therapeutic or imaging agent may be conjugated, fused to, or otherwise attached to the antibody.

The antibodies and nanoparticles may be used as carriers of a therapeutic or imaging agent, including a chemotherapeutic agent, such as doxorubicin, or a prodrug, such as irinotecan (CPT-11). The therapeutic or imaging agent may be a recombinant protein or a peptide. The therapeutic agent may be a toxin, such as botulinum toxin. The therapeutic or diagnostic agent may be an engineered nucleic acid, such as a therapeutic RNA or an aptamer. Any anti-tumor therapeutic agent or imaging agent known in the art may be used, coupled, conjugated, entrapped, or encapsulated by/to an antibody or nanoparticle. An antibody may be a therapeutic agent on its own, without coupling to another moiety.

Examples of types of therapeutic agents and specific examples include, without limitation, alkylating antineoplastic agents, such as cisplatin and carboplatin, oxaliplatin, mechlorethamine, cyclophosphamide, chlorambucil, ifosfamide, antimetabolites such as azathioprine, mercaptopurine, alkaloids, such as vinca alkaloids and taxanes, vincristein, binblastine, vinorelbine, vindesine, podophyllotoxin, docetaxel, topoisomerase inhibitors such as topotecan. amsacrine, etoposide, etoposide phosphate, and teniposide, cytotoxic antibiotics, such as actinomycin, anthracyclines, doxorubicin, daunorubicin, valrubicin, idarubicin, epirubicin, bleomycin, plicamycin, and mitomycin.

A non-limiting list of toxins which may be used as a therapeutic agent include Abrin, Aerolysin, Botulinm toxin A, Botulinm toxin B, Botulinin toxin C1, Botulinin toxin C2, Botulinin toxin D, Botulinin toxin E, Botulinin toxin F, b-bungaretoxin, Caeruleotoxin, Cereolysin, Cholera toxin, Clostridium difficile enterotoxin A, Clostridium difficile cytotoxin B, Clostridium perfringens lecithinase, Clostridium perfringens kappa toxin, Clostridium perfringens perfringolysin O, Clostridium perfringens enterotoxin, Clostridium perfringens beta toxin, Clostridium perfringens delta toxin, Clostridium perfringens epsilon toxin, Comotoxin, Crotoxin, Diphtheria toxin, Listeriolysin, Leucocidin, Modeccin, Nematocyst toxins, Notexin, Pertussis toxin, Pneumolysin, Pseudomonas aeruginosa toxin A, Ricin, Saxitoxin, Shiga toxin, Shigella dysenteriae neurotoxin, Streptolysin O, Staphylococcus enterotoxin B, Staphylococcus enterotoxin F, Streptolysin S, Taipoxin, Tetanus toxin, Tetrodotoxin, Viscumin, Volkensin, and Yersinia pestis murine toxin,

A detectable imaging agent can be coupled, conjugated, entrapped, or encapsulated by/to an antibody or nanoparticle. The imaging agent may be a magnetic material, a photosensitizing agent, a contrast agent, or a radionuclide, for example. The radionuclide may be, for example, Iodine-131 (131I), Iodine-125 (125I), Fluorine-18 (18F), Gallium-68 (68Ga), Copper-64 (64Cu), Copper-67 (67Cu), Zirconium-89 (89Zr), Yttrium-90 (90Y), Lutetium-177 (177Lu), Indium-111 (111In), or Technetium-99m (99mTc). Contrast imaging agents for Magnetic Resonance Imaging (MRI) may include any known in the art including a gadolinium-based contrast agent. Other imaging agents which may be used include Feridex I.V., mangafodipir (Teslascan), a contrast agent for ultrasound, such as a micro-bubble contrast agent, and fluorodeoxyglucose (18F). After the imaging agent is administered and a suitable time is elapsed for the agent to reach the target tumor, a non-invasive detection technique is performed to generate an image of the tumor in the individual. Suitable techniques include without limitation MRI, ultrasound, PET, and CT scan.

Vascular-active permeability agents are those which increase the amount of a therapeutic or imaging agent which is delivered via the circulation to a tumor. Without limiting the invention to any particular mechanism of action, the agents may act by causing vascular inflammation, or by disrupting the vasculature so that agents of a size which were previously not delivered are delivered, or so that an increase in the amount of an agent of a certain size is delivered. Exemplary vascular-active permeability agents include bacteria (including bacteria which spontaneously infect tumors), such as Clostridium novyi-NT, bacterial spores, a bacterial component, for example lipopolysaccharide (LPS), a vaccine, Coley's toxin, a cytokine, such as tumor necrosis factor-alpha (TNF-α), interferon-gamma (IFN-γ), or interleukin-2 (IL-2), a chemokine, an inducer of cytokine or chemokine expression, e.g., vadimezan (ASA404, DMXAA), and inducer of vascular inflammation, an immune response modifier, a hormone, a pressor agent, such as angiotensin II or adrenaline, a virus, a microtubule interacting agent, such as vinorelbine, combretastatin A4 phosphate (CA4P), BTI-286, or colchicine, a nitric oxide synthase inhibitor, such as L-NAME, L-NNA, or L-NMMA, tumor-localized radiation, and tumor-localized thermotherapy, and high intensity focused ultrasound.

Any combination of named therapeutic or imaging agents with vascular-active permeability entities are specifically contemplated as if each combination were listed separately and explicitly.

Any method known in the art can be used to determine whether the amount of therapeutic agent or imaging agent delivered is increased. These include, without limitation in vivo imaging, biopsy, and agent localization, or tumor response using RESIST criteria. The (1) vascular-active permeability agent and (2) nanoparticles or antibodies with (3) an imaging agent can be used to assess appropriateness of treatment or appropriate dosages of (a) the vascular-active permeability agents and (b) nanoparticles or antibodies with (c) a therapeutic agent. Thus these can be used sequentially or iteratively. Kits may contain the reagents for both assessment and therapeutic uses.

Solid tumors to be treated may be of any type and in any organ of the body of a mammal, such as a farm animal, a pet, a laboratory animal, or a human. These may be in the brain, colon, breasts, prostate, liver, kidneys, lungs, esophagus, head and neck, ovaries, cervix, stomach, colon, rectum, bladder, uterus, testes, and pancreas, as non-limiting examples. The type of tumor may be an adenocarcinoma, a squamous cell carcinoma, or a sarcoma, for example.

The major limitation for most chemotherapeutic agents is their toxicity toward normal tissues, which prohibits the use of doses high enough to eradicate all cancer cells. One approach to address this problem is to develop agents that are delivered to all cells but are preferentially toxic to tumor cells because of the abnormal signaling pathways. This strategy underlies the success of agents such as Gleevec (imatinib) and Iressa (gefitinib) (26, 27). A second approach is to use agents that bind to extracellular molecules present at higher concentrations on the surface of tumor cells, such as Herceptin (trastuzumab) and Erbitux (cetuximab) (28, 29). The third approach takes advantage of the abnormal vasculature present in tumors, allowing preferential accumulation of nanoparticles (the EPR effect) (6, 30). Though all approaches have merit, the third has the advantage that virtually any drug, including a wealth of clinically approved agents, can in theory be made more effective by its incorporation into nanoparticles of appropriate sizes. The ability to use agents that are already clinically approved poses many practical advantages with respect to the performance of clinical trials and the duration of the drug approval process.

In this work, we have attempted to enhance the third approach through pharmacologic manipulation of the abnormal vasculature already present in tumors. We show that Enhanced EPR (E2PR). can dramatically increase the tumor: blood ratio of nanoparticles, as assessed by both imaging and therapeutic response. It is worth noting that even a small difference in the intratumoral concentration of an agent can make a large difference in therapeutic effect (31). In the studies described here, E2PR led to a tumor: blood ratio of more than 22-fold (FIG. 2A).

We were particularly encouraged with the results in the GBM model. This tumor type in humans is highly recalcitrant to conventional therapies, leading to a dismal prognosis for patients with this disease. The blood-brain barrier is at least partly to blame for the limited efficacy of chemotherapy (32). We found that TNF-α treatment could help breach the blood-brain barrier and result in major accumulations of 125I-labeled sterically stabilized liposomal nanoparticles (SSLs) in the orthotopically implanted brain tumors as well as significantly prolong the survival of the tumor-bearing animals (FIG. 4). As the mouse cranial cavity is small, murine brain tumors are particularly difficult to treat as even a minimal amount of growth of a pre-existing tumor is lethal.

Our results suggest a way to improve the therapeutic efficacy of conventional and novel drugs by incorporating them into nanoparticles and injecting them together with vascular-active agents such as TNF-α. The approach is versatile, as it should be practicable with a variety of nanoparticle formulations as well as with diverse chemical and radioactive agents.

The above disclosure generally describes the present invention. All references disclosed herein are expressly incorporated by reference. A more complete understanding can be obtained by reference to the following specific examples which are provided herein for purposes of illustration only, and are not intended to limit the scope of the invention.

Figure 10: Qualitative case study illustrating the ground-truth reference patent.

H Discussion on Human Evaluation and Legal Adequacy

While this work focuses on algorithmic mechanisms for controllable, evidence-grounded long-form patent drafting in an early-stage drafting setting, we emphasize that the proposed framework is intended as a structured drafting assistant rather than a replacement for professional patent attorneys. Accordingly, aspects critical to real-world patent practice, such as legal adequacy, claim dependency correctness, and antecedent basis consistency, ultimately require expert human judgment. These criteria extend beyond surface-level textual quality and factual alignment, involving nuanced legal interpretation and jurisdiction-specific drafting conventions that remain difficult to reliably capture with automatic metrics.

We view human evaluation by patent practitioners as an essential direction for future validation. A principled evaluation protocol would involve expert assessment of generated drafts along dimensions including (i) legal adequacy of technical disclosure, (ii) correctness and consistency of claim dependencies, and (iii) validity of antecedent basis usage, potentially complemented by qualitative feedback on drafting clarity and completeness. We believe that the logic-tree-based intermediate representation introduced in this work provides a favorable foundation for such evaluations by explicitly exposing conceptual dependencies and evidence grounding, thereby facilitating systematic inspection and correction by human experts.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims are supported by both quantitative and qualitative evaluations. As shown in Table 1 and Figures 3-6, LogicTree-RAG consistently improves content quality, factuality, and structural coherence over strong baselines.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper explicitly discusses limitations in Section 4.5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.

- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This work is primarily empirical and systems-oriented, focusing on algorithmic design and evaluation rather than formal theoretical guarantees.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides: (i) detailed dataset descriptions (Pap2Pat, LCFO); (ii) full metric definitions (Appendix F.1); (iii) baseline details (Appendix F.2); (iv) hyperparameters and decoding settings (Appendix F.3); (v) prompts and implementation details (Appendix E, F.5). These ensure reproducibility.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but

reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The implementation details are fully specified, and code release is planned to facilitate reproducibility (to be included upon acceptance).

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper provides hyperparameters and decoding settings in Appendix F.3 and provides prompts and implementation details in Appendices E and F.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We use expert evaluation to validate the significance of our method in the real world.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper specifies (i) framework implementation (PyTorch, Milvus) in Section 4.1; (ii) model settings and inference configuration in Appendices F.3 and F.4; (iii) computational cost analysis in Appendix G.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We obey the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have discussed the broader impacts in Section 4.5 and Appendix H.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This work does not pose such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have cited the corresponding assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This work does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.