STRUCTGEO: INTENT-GUIDED STRUCTURAL ORGANIZATION FOR GENERATIVE ENGINE OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative Engine Optimization (GEO) aims to rewrite a document so that generative engines are more likely to include its content in their responses. Recent studies infer potential user intents and then use them to guide rewriting. While effective, they have two limitations. First, because potential queries are inferred independently, they tend to concentrate on a few dominant search intents, resulting in insufficient coverage of less prominent information needs. Second, they focus on what evidence a document contains but overlook how that evidence is organized, even though organization affects whether a generative engine actually uses it. To address both issues, we propose **StructGEO**, a GEO framework that models search intents hierarchically and organizes supporting evidence to match. Specifically, StructGEO first constructs a *top-down intent tree* by recursively decomposing broad search intents into fine-grained ones. It then performs *structure-aware rewriting* by consolidating evidence relevant to similar intents into coherent sections. These steps broaden the coverage of information needs and keep related evidence coherent rather than scattered throughout the document. Extensive experiments show that StructGEO consistently outperforms existing GEO methods, achieving average gains of 21.2% and 17.5% over AutoGEO and IF-GEO, respectively, in Position-Adjusted Word Count (Word & Pos) visibility.

1 INTRODUCTION

Generative engines (Liu et al., 2023; Kirsten et al., 2026) are search systems that combine document retrieval with large language models (LLMs). Given a user query, they retrieve relevant documents and generate a natural-language answer that synthesizes information from them. Whereas traditional search engines (Brin & Page, 1998; Liu, 2010) return a ranked list of documents, generative engines integrate the retrieved evidence through retrieval-augmented generation (RAG) (Lewis et al., 2020; Guu et al., 2020; Izacard & Grave, 2021) and explicitly cite the documents they draw on. As generative engines become widely adopted, *Generative Engine Optimization (GEO)* (Aggarwal et al., 2024; Wu et al., 2026b; Zhou et al., 2026; Kim et al., 2026) has emerged as an important research problem. GEO aims to improve a document's visibility in generated responses by rewriting its content so that generative engines are more likely to use and cite it.

Early GEO methods mostly relied on manually designed heuristics (Aggarwal et al., 2024; Kim et al., 2026) or on rewriting strategies learned from visibility differences across documents (Wu et al., 2026b). However, neither approach explicitly considers what information users might seek from a given document. To fill this gap, recent work (Zhou et al., 2026) has proposed a document-specific method that infers the queries that the user might issue for a document, treats them as proxies for the document's information needs, and uses them to guide rewriting. Because the queries are tailored to each document, the rewritten document can better serve users' needs.

Despite this progress, existing methods still face two key challenges, as illustrated in Figure 1. First, the inferred queries tend to concentrate on only a few information needs. Because existing methods provide no explicit mechanism for covering diverse search intents, the inferred queries may miss important aspects of the document. As a result, query-guided rewriting can overemphasize specific aspects while leaving other potential information needs insufficiently covered. Second, prior work does not consider how the organization of relevant evidence within a document affects whether generative engines actually use it. Recent studies in RAG settings (Liu et al., 2024; Tian et al., 2025)

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

have shown that LLMs struggle to utilize relevant evidence when it is dispersed across the context, and the same issue can arise when evidence is scattered within a single document. (We analyze both limitations in detail in Section 3.)

To address these issues, we propose **StructGEO**, a GEO framework that hierarchically models potential information needs and organizes their supporting evidence into coherent structures. The key idea is twofold. StructGEO first uncovers search intents progressively, from broad to specific, so that the inferred queries cover a wider range of information needs. It then gathers the evidence for related intents into the same part of the rewritten document. Concretely, StructGEO operates in two stages:

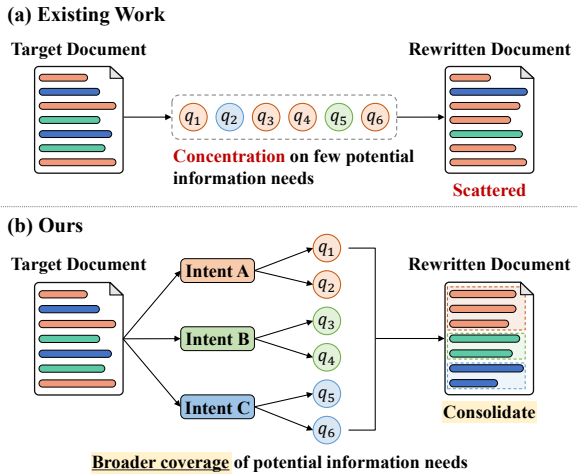


Figure 1: Illustration of our motivation. While (a) existing GEO methods independently infer potential queries without considering search intent coverage or evidence organization, (b) StructGEO hierarchically models information needs and organizes related evidence into coherent sections.

(i) Top-down intent tree construction. Given a target document, StructGEO first identifies coarse-grained search intents that capture major aspects. It then recursively decomposes each coarse-grained intent into fine-grained ones, conditioning each child intent on its parent. Because intents are derived within this hierarchy, the inferred queries do not collapse onto a few dominant intents, yielding a broader range of potential information needs.

(ii) Structure-aware rewriting. StructGEO then rewrites the document along the constructed intent tree, from the leaves up to the root. Each leaf node rewrites the source evidence associated with its fine-grained intent; each internal node integrates the outputs of its children into a coherent section centered on the parent intent; and the root node combines the resulting top-level sections, together with any remaining source information, into the final document. Because evidence for related intents is consolidated within each subtree, relevant evidence stays locally coherent in the rewritten document, making it easier for generative engines to utilize.

In this way, StructGEO jointly addresses *what* information needs should guide document rewriting and *how* the supporting evidence should be organized. Extensive experiments on two datasets, **GEO-Bench** (Aggarwal et al., 2024) and **E-Commerce** (Wu et al., 2026b), using generative engines built on three state-of-the-art LLMs, show that StructGEO consistently outperforms existing methods, achieving average Word & Pos improvements of 21.2% over AutoGEO (Wu et al., 2026b) and 17.5% over IF-GEO (Zhou et al., 2026). These results highlight the benefits of hierarchically modeling information needs and coherently organizing the evidence for related search intents.

Our contributions are summarized as follows:

- We introduce hierarchical intent modeling for GEO, which recursively decomposes broad search intents into finer-grained ones to prevent the inferred queries from concentrating on a few intents and broaden the coverage of potential information needs.
- We develop a structure-aware rewriting framework for GEO that, rather than relying on predefined structural features or rules, derives document-specific organization from the hierarchy of potential information needs and consolidates evidence for relevant search intents into coherent sections.
- Extensive experiments on two datasets with three state-of-the-art LLM backbones show that StructGEO consistently improves document visibility over existing GEO methods.

2 Related Work

Generative engine optimization (GEO). GEO (Aggarwal et al., 2024; Puerto et al., 2025; Kim et al., 2026) aims to improve document visibility in generative-engine responses through content rewriting. Early work (Aggarwal et al., 2024) relied on manually designed strategies, while subsequent

studies explored generative-engine preferences (Wu et al., 2026b), feature-level optimization (Liu & Xu, 2026), and reusable strategies (Wu et al., 2026a). More recent methods tailor optimization to instance-specific information needs. IF-GEO (Zhou et al., 2026) infers potential queries for a document, whereas Mind Reader (Chen et al., 2026) derives potential user demands from a given real user query. However, IF-GEO treats inferred queries as a flat set, which may overrepresent few dominant search intents. In contrast, our work organizes potential information needs hierarchically to cover broader aspects of a document.

Context position and information distance. Even when relevant evidence is present in the context, LLMs do not always utilize it effectively. Prior work (Liu et al., 2024; Hsieh et al., 2024; He et al., 2024) shows that RAG performance depends on evidence position in the context, a phenomenon known as *lost-in-the-middle*. Beyond absolute position, recent studies (Tian et al., 2025; Wang et al., 2025) show that greater distance between related pieces of evidence can also degrade performance. In particular, the *lost-in-the-distance* phenomenon (Wang et al., 2025) shows that intervening irrelevant text between query-related pieces can hinder their joint utilization. However, whether this phenomenon also influences document visibility in GEO remains largely unexplored.

Recent GEO methods have begun to consider document structure, but existing approaches mainly rely on predefined structural features or rules. FeatGEO (Liu & Xu, 2026) optimizes hand-designed features such as heading hierarchy and list density, while GEO-SFE (Yu et al., 2026) applies empirically derived structural rules. In contrast, StructGEO derives document organization from a document-specific hierarchy of inferred search intents and uses this hierarchy to consolidate relevant evidence into coherent sections. To the best of our knowledge, StructGEO is the first GEO framework to organize rewritten documents based on a document-specific hierarchy.

3 Motivating Analysis

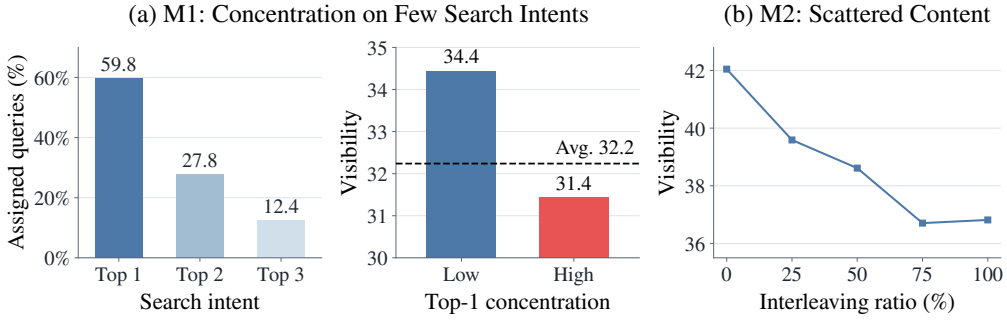


Figure 2: Motivating Analysis of the two limitations addressed by StructGEO on E-Commerce. **(a) M1: Concentration of independently inferred queries on few potential information needs.** *Left:* 59.8% of queries inferred by IF-GEO are concentrated on a single search intent. *Right:* The high-concentration group exhibits lower visibility than the low-concentration group. **(b) M2: Effect of Evidence Cohesion on Visibility.** The interleaving ratio ranges from 0% (contiguous) to 100% (maximally dispersed). Dispersion of relevant evidence can reduce document visibility even when the underlying information remains unchanged.

Motivation 1 (M1): Independently inferred queries concentrate on a few information needs. We first examine whether the potential queries inferred by existing document-specific GEO methods are biased toward a few search intents. For each target document, we use an LLM to identify broad search intents corresponding to its major content aspects. We then apply IF-GEO (Zhou et al., 2026) to independently generate 15 potential queries for the document and ask an LLM to assign each query to one of the identified intents. We rank the intents by the number of assigned queries and define each intent's *query share* as the fraction of queries assigned to it. Figure 2(a, left) shows that the top-ranked intent receives 59.8% of the queries on average, whereas the second- and third-ranked intents receive only 27.8% and 12.4%, respectively. To examine whether this concentration is associated with GEO performance, we split the instances into a *high-concentration* group, in which at least 50% of the queries fall under the top-ranked intent, and a *low-concentration* group, which contains the remaining instances. As shown in Figure 2(a, right), the high-concentration group exhibits lower visibility (Word

& Pos) than the low-concentration group (31.4 vs. 34.4). These results motivate us to model potential information needs hierarchically to encourage broader coverage across diverse search intents by allocating the query budget more evenly across broad intents.

Motivation 2 (M2): Coherently organized evidence improves document visibility. We next examine how the organization of relevant evidence affects document visibility when the document's content is held fixed. Starting from a document rewritten by StructGEO, we locate the section corresponding to the broad intent that matches the real user query and place it at the center of the document to control for positional bias. We then progressively interleave sentences from other sections between the sentences of this query-relevant section. We quantify the degree of dispersion by the *interleaving ratio*, which ranges from 0% when the section remains contiguous to 100% when its sentences are maximally dispersed; the document content is unchanged throughout. As shown in Figure 2(b), the Word & Pos score decreases steadily as the interleaving ratio increases. This result motivates us to consolidate evidence for related search intents into coherent sections.

4 Proposed Method: StructGEO

4.1 Problem Formulation

Following prior work (Aggarwal et al., 2024; Wu et al., 2026b; Zhou et al., 2026), we consider a RAG-based generative engine. Given a user query q, the generative engine G retrieves a set of relevant candidate documents D_q and uses an LLM to generate a response $r_q = G(q, D_q)$. The generated response r_q may synthesize information from multiple documents in D_q and cite them as supporting sources. The visibility function $v_G(d_i, q)$ quantifies the contribution of a document $d_i \in D_q$ to the response generated for query q.

The task of GEO is to rewrite a target document d_T to improve its visibility in the generative engine's response. A GEO method M transforms d_T into a rewritten document $d'_T = M(d_T)$, and the visibility gain of d'_T for query q is defined as follows:

$$\Delta v_G(d'_T, d_T, q) = v_G(d'_T, q) - v_G(d_T, q). \tag{1}$$

Accordingly, the objective of GEO is to develop a rewriting strategy that maximizes the expected visibility gain across a set of target documents $\mathcal{D}_T$ and their associated query sets.

$$\arg\max_M \frac{1}{|\mathcal{D}_T|} \sum_{d_T \in \mathcal{D}_T} \frac{1}{|\mathcal{Q}(d_T)|} \sum_{q \in \mathcal{Q}(d_T)} \Delta v_G(d'_T, d_T, q), \tag{2}$$

where $\mathcal{Q}(d_T)$ denotes the set of user queries associated with the target document d_T.

However, at optimization time, the set of user queries associated with the target document, $\mathcal{Q}(d_T)$, is not observable, and the generative engine is accessible only as a black box (Aggarwal et al., 2024). Therefore, a GEO method should infer potential information needs solely from the target document.

4.2 Overview of StructGEO

As illustrated in Figure 3, StructGEO aims to broaden the coverage of potential information needs while organizing evidence relevant to similar search intents into coherent structures in the rewritten document. To achieve these goals, StructGEO consists of two main stages. First, we recursively construct an intent tree to hierarchically model potential information needs associated with the target document. Then, following the structure of the intent tree, we perform structure-aware rewriting in a bottom-up manner to produce the final document d'_T.

4.2.1 Top-down Intent Tree Construction

This stage aims to broaden the coverage of potential information needs by preventing potential queries from concentrating on few dominant information needs. To this end, we first infer broad search intents that capture the major aspects of the target document and recursively derive more fine-grained search intents conditioned on their parent intents to construct an intent tree. Specifically, StructGEO allocates the same query budget to each broad intent, preventing a small subset of intents from dominating the inferred queries. Each node is grounded in relevant source evidence from the target document.

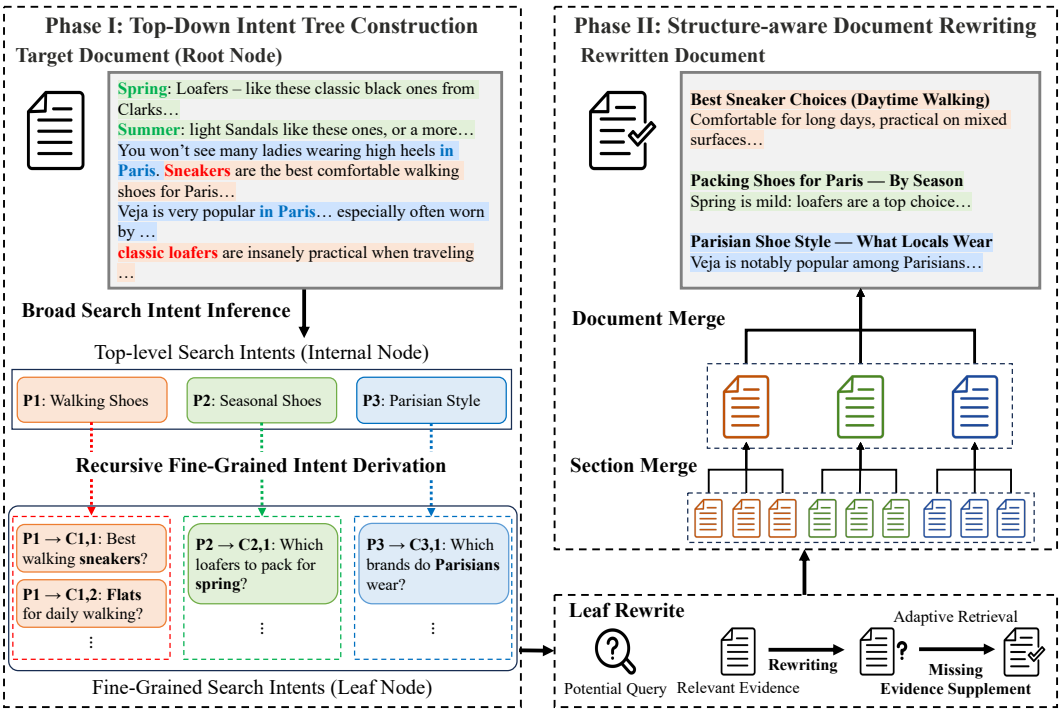


Figure 3: Overview of StructGEO. StructGEO consists of two main stages. First, it constructs an intent tree to hierarchically model potential information needs associated with the target document. Then, it performs structure-aware rewriting using the intent tree as a structural guide.

Intent tree. Given a target document d_T, we introduce a root node v_{root} representing the entire document and recursively construct an intent tree in a top-down manner. Each non-root intent node v is represented by three components: a potential search query z_v, an intent label l_v, and an intent type t_v. The potential search query expresses the information need captured by the node as a natural-language query, the intent label provides a concise summary, and the intent type specifies the type of information requested by the query. For example, a node with the potential search query "How long does the battery last on a full charge?" can be associated with the intent label "Battery Life", and the intent type "product_runtime".

Let L denote the number of levels in the intent tree, with the root node at depth 0 and the leaf nodes at depth $L-1$. The LLM first generates a set of child intent nodes $\mathcal{C}_{v_{\text{root}}}$ conditioned on the target document d_T. At subsequent depths, for each non-leaf internal node v, the LLM generates a set of child intent nodes $\mathcal{C}_v$ conditioned on the intent represented by (z_v, l_v, t_v). Intents at shallower depths capture broad information needs associated with the document, while successive recursive expansions derive more fine-grained information needs conditioned on their parent intents. This process continues until depth $L-1$ is reached, where each leaf node corresponds to a concrete and plausible potential search query that a real user could ask about the document.

Formally, let $\mathcal{V}^{(\ell)}$ denote the set of nodes at depth ℓ, and let $\mathcal{C}_v$ denote the set of child intent nodes generated from node v, where $\mathcal{C}_v = \varnothing$ for a leaf node. Starting from $\mathcal{V}^{(0)} = \{v_{\text{root}}\}$, the nodes at each subsequent depth are recursively defined as follows:

$$\mathcal{V}^{(\ell+1)} = \bigcup_{v \in \mathcal{V}^{(\ell)}} \mathcal{C}_v, \qquad 0 \le \ell \le L-2. \tag{3}$$

For an intent tree with L levels, the resulting intent tree $\mathcal{T}_{d_T} = (\mathcal{V}, \mathcal{E})$ is defined as follows:

$$\mathcal{V} = \bigcup_{\ell=0}^{L-1} \mathcal{V}^{(\ell)}, \qquad \mathcal{E} = \bigcup_{\ell=0}^{L-2} \left\{ (v, c) \mid v \in \mathcal{V}^{(\ell)}, c \in \mathcal{C}_v \right\}. \tag{4}$$

Unless otherwise specified, we construct an intent tree with $L = 3$ levels.

Grounding node evidence. After constructing the intent tree $\mathcal{T}_{d_T}$, we segment the target document into an ordered sequence of source segments, $S(d_T) = (s_1, \ldots, s_N)$, where N denotes the number of source segments. We preserve paragraph boundaries whenever possible and split long paragraphs at sentence boundaries to obtain localized evidence units. Details of the segmentation procedure are provided in Appendix G.1.

Let $f(\cdot)$ denote a dense text encoder. For each non-root intent node $v \in \mathcal{V} \setminus \{v_{\text{root}}\}$, we compute the relevance between its potential search query z_v and each source segment s_i as follows:

$$R(v, s_i) = \cos\left(f(z_v), f(s_i)\right). \tag{5}$$

For each non-root intent node v, we assign a set of source evidence A_v consisting of the top-k source segments whose relevance scores exceed a threshold τ:

$$A_v = \underset{s_i \in S(d_T)}{\text{Top}_k} \; R(v, s_i) \quad \text{s.t.} \quad R(v, s_i) \geq \tau. \tag{6}$$

Because a single source segment may support multiple related intents, we allow the same segment to be assigned to multiple intent nodes. If no segment satisfies the relevance threshold for a non-root node, we remove the node and its descendant subtree from the intent tree. For the root node, which does not have a potential search query, we use the embedding of the entire target document as the root representation and apply the same evidence assignment procedure to obtain $A_{v_{\text{root}}}$. We use Qwen3-Embedding-8B (Zhang et al., 2025) as the text encoder $f(\cdot)$, with $k = 3$ and $\tau = 0.20$.

4.2.2 Structure-aware Document Rewriting

StructGEO rewrites the target document bottom-up along the intent tree, using it as a structural guide to reduce the dispersion of evidence relevant to similar search intents: (i) Each leaf node first produces an intent-specific rewrite from its assigned source evidence. (ii) Internal nodes then merge their child outputs into coherent sections centered on the corresponding parent intents, and (iii) the root node integrates the resulting top-level sections with remaining source information into the final document. This structure-aware rewriting process keeps related evidence locally grouped.

Leaf rewrite. The leaf rewrite stage tailors source evidence to the fine-grained information need of each leaf intent. Given its potential search query z_{v_l} and source evidence A_{v_l}, the LLM identifies up to five shortfalls in A_{v_l} with respect to z_{v_l} and converts them into actionable rewriting rules specifying what should be clarified, reorganized, or elaborated. It then applies these rules to produce an initial rewritten passage $\tilde{h}_{v_l}$:

$$\tilde{h}_{v_l} = \text{Rewrite}(A_{v_l} \mid z_{v_l}). \tag{7}$$

Because A_{v_l} may omit relevant details scattered elsewhere in the document (*e.g.*, numerical values or supporting explanations), we employ adaptive retrieval (Jiang et al., 2023; Jin et al., 2025; Li et al., 2025) to supplement the initial rewrite. Given z_{v_l}, $\tilde{h}_{v_l}$, and A_{v_l}, the LLM identifies missing or insufficiently explained information and generates a retrieval query for each need. For each query, we retrieve relevant segments from $S(d_T)$ using the same dense retrieval procedure, excluding those already included in A_{v_l}, and pair each need with its supporting evidence to form $\mathcal{P}^+_{v_l}$. The LLM then revises the initial rewrite to produce the final leaf output h_{v_l}:

$$h_{v_l} = \text{Revise}\left(\tilde{h}_{v_l}, \mathcal{P}^+_{v_l} \; ; \; z_{v_l}, A_{v_l}\right). \tag{8}$$

If no missing information need or supporting evidence is identified, we set $h_{v_l} = \tilde{h}_{v_l}$.

Section merge. To reduce evidence dispersion, each internal node v integrates its non-empty child outputs into a coherent section conditioned on the corresponding parent intent (z_v, l_v, t_v) and its evidence A_v:

$$h_v = \text{SectionMerge}\left(\{h_c \mid c \in \mathcal{C}_v, \; h_c \neq \varnothing\} \mid (z_v, l_v, t_v), A_v\right). \tag{9}$$

This process consolidates evidence within each subtree before combining it with content from other subtrees. Rather than simply concatenating the child outputs, the model reorganizes them around the parent intent, preserving the distinct information contributed by each child and removing irrelevant content, while using A_v to resolve potential inconsistencies. The resulting h_v is passed to the parent as a section draft.

Document merge. After all internal-node merges, the root integrates the top-level outputs into the final document while incorporating unassigned source evidence to avoid information loss. Using the root evidence $A_{v_{\text{root}}}$ and a set of unassigned evidence segments U, the model removes redundant content and improves transitions, coherence, and narrative flow while preserving section boundaries:

$$d'_T = \text{DocumentMerge}\left(\{h_c \mid c \in \mathcal{C}_{v_{\text{root}}},\ h_c \neq \varnothing\}\ ;\ U \mid A_{v_{\text{root}}}\right). \tag{10}$$

where d'_T denotes the final rewritten document.

5 Experimental Setup

Datasets. Following prior work (Aggarwal et al., 2024; Zhou et al., 2026; Wu et al., 2026b), we evaluate our method on two datasets, **GEO-Bench** and **E-Commerce**. GEO-Bench (Aggarwal et al., 2024) contains 10K samples covering diverse open-domain queries (Kwiatkowski et al., 2019; Craswell et al., 2021; Alexander et al., 2022; Zhou et al., 2023; Liu et al., 2023), with 8K, 1K, and 1K samples for the training, validation, and test splits, respectively. E-Commerce (Wu et al., 2026b) consists of 1,667 training queries and 416 test queries curated from e-commerce-related conversations in LMSYS-CHAT-1M (Zheng et al., 2024). Each sample consists of a user query, five corresponding candidate documents retrieved by Google Search, and the index of the target document. For evaluation, we replace only the target document with its rewritten version produced by each GEO method while leaving the remaining candidate documents unchanged.

Baselines. *Vanilla* denotes the performance of the original document without applying any GEO method. We compare StructGEO with existing GEO methods, which we categorize into three groups. First, the heuristic approach uses manually designed rewriting strategies (Aggarwal et al., 2024); we adopt *Statistics Addition*, *Authoritative*, and *Fluency Optimization*. Second, the global-preference approach infers rewriting rules that can be applied across multiple documents and uses them for optimization. We use AutoGEO (Wu et al., 2026b) as the representative baseline in this category. Third, the document-specific approach constructs rewriting strategies tailored to individual documents based on their content and potential information needs. We use IF-GEO (Zhou et al., 2026) as the representative baseline in this category. To control for the number of potential queries that directly induce intent-specific rewrites, we set the number of potential queries in IF-GEO equal to the number of leaf nodes in StructGEO. We also provide an analysis of the LLM-call complexity of IF-GEO and StructGEO, along with a call-matched performance comparison, in Appendix A. The prompts used in each stage of StructGEO are provided in Appendix J.

For all methods, we use GPT-5 mini (OpenAI, 2026), Gemini-3.1 Flash-Lite, and Qwen3.8-27B as backbone LLMs for both the document optimizer and the generative engine. Other implementation details are provided in Appendix G.2.

Evaluation metrics. To evaluate the visibility of the target document, we use three objective visibility metrics (Aggarwal et al., 2024): *Word*, *Pos*, and *Word & Pos* (*i.e.*, Overall). *Word* measures the proportion of tokens in sentences that cite the target document relative to the entire response, while *Pos* measures how early the target document is cited in the response. *Word & Pos* captures document visibility by jointly considering both factors.

6 Experimental Results

6.1 Overall Performance

Table 1 reports the performance of StructGEO and five GEO baselines across two datasets and three LLM backbones. **StructGEO consistently achieves the best performance across all datasets and backbone models**, yielding average Word & Pos gains of 21.2% and 17.5% over AutoGEO and IF-GEO, respectively. Consistent improvements in both Pos and Word indicate that the gains are not confined to a single component of visibility. Moreover, averaged across the two datasets, StructGEO improves Word & Pos over IF-GEO by 14.9%, 11.2%, and 26.3% with GPT-5 mini, Gemini-3.1 Flash-Lite, and Qwen3.8-27B, respectively, demonstrating robustness across LLM backbones. These results highlight the benefits of hierarchically modeling potential information needs and explicitly organizing their supporting evidence beyond document-specific query-guided rewriting alone. We further show

Table 1: Performance comparison of StructGEO with five GEO baseline methods on GEO-Bench and E-Commerce datasets. The best performance is marked in **bold**, and the second-best is underlined. * indicates a statistically significant improvement of StructGEO over the best baseline in the corresponding column, based on a two-sided paired t-test over query-level scores averaged across the three runs ($p < 0.001$).

Type	Method	GPT-5 mini			Gemini-3.1 Flash-Lite			Qwen3.8-27B		
		Pos	Word	**Overall**	Pos	Word	**Overall**	Pos	Word	**Overall**
GEO-Bench										
Original	Vanilla	19.85	19.91	19.96	19.93	19.80	19.94	19.72	19.75	19.73
Heuristic	Authoritative	21.39	21.47	21.50	21.81	21.48	21.67	20.26	20.24	20.27
	Statistics Addition	22.33	22.65	22.50	22.27	22.10	22.20	23.23	23.29	23.19
	Fluency Optimization	22.88	22.93	22.93	19.28	19.12	19.21	20.77	20.99	20.89
Global Preference	AutoGEO	33.12	33.74	33.40	27.62	27.41	27.52	30.97	30.74	31.12
Document-specific	IF-GEO	33.32	34.00	33.69	33.69	33.79	33.59	31.51	31.59	31.71
	StructGEO	**38.35***	**39.44***	**38.72***	**35.65***	**36.06***	**35.76***	**38.26***	**38.69***	**38.54***
E-Commerce										
Original	Vanilla	18.07	17.93	18.04	17.77	17.78	17.78	18.92	19.01	18.87
Heuristic	Authoritative	18.55	18.67	18.66	21.15	20.90	21.02	20.01	19.86	20.13
	Statistics Addition	19.81	20.01	19.84	19.85	19.84	19.84	21.09	20.92	21.06
	Fluency Optimization	20.98	21.25	21.22	18.99	19.10	19.10	21.28	21.31	21.36
Global Preference	AutoGEO	33.89	34.27	34.30	29.42	30.28	29.54	33.34	33.12	33.75
Document-specific	IF-GEO	32.46	32.36	32.57	32.99	33.63	33.05	30.39	30.34	30.58
	StructGEO	**36.88***	**38.19***	**37.40***	**38.47***	**39.45***	**38.29***	**39.74***	**40.00***	**40.10***

in Appendix B that these visibility gains are accompanied by improvements in the overall quality of the rewritten documents, and in Appendix C that they cannot be explained solely by document expansion. Appendix F further illustrates how StructGEO reorganizes information associated with similar intents into more coherent sections, in contrast to the more scattered organization observed in competing methods.

6.2 Cross-backbone Evaluation

Recent studies (Panickssery et al., 2024; Spiliopoulou et al., 2025) have shown that LLMs can exhibit self-preference bias, favoring outputs generated by themselves or models from the same family. To examine whether GEO performance is affected by such backbone-specific preferences, we evaluate three GEO methods (*i.e.*, AutoGEO, IF-GEO, and StructGEO) in cross-backbone settings, where the optimizer and generative engine use different LLMs. Table 2 reports the results on E-Commerce.

Table 2: Cross-backbone evaluation on E-Commerce. Document optimizer backbones are grouped by rows, while generative engine backbones are shown in columns.

Optimizer	Method	Generative Engine			Avg.
		GPT-5	Gemini-3.1	Qwen3.8	
GPT-5	AutoGEO	34.30	34.19	27.21	31.90
	IF-GEO	32.57	33.06	**30.60**	32.08
	StructGEO	**37.40**	**36.65**	28.55	**34.20**
Gemini-3.1	AutoGEO	23.70	29.54	26.98	26.74
	IF-GEO	26.66	33.05	30.49	30.07
	StructGEO	**28.68**	**38.29**	**33.07**	**33.35**
Qwen3.8	AutoGEO	28.81	34.25	33.75	32.27
	IF-GEO	27.47	30.83	30.58	29.63
	StructGEO	**35.21**	**40.77**	**40.10**	**38.69**

(i) Self-preference bias in GEO. All three methods perform better on average in same-backbone than in cross-backbone settings, with relative Word & Pos gains of 11.4%, 7.4%, and 14.1% for AutoGEO, IF-GEO, and StructGEO, respectively. This pattern is consistent with self-preference bias, suggesting that such bias may influence GEO evaluation. However, some cross-backbone combinations outperform their corresponding same-backbone settings, indicating that this effect is not universal.

(ii) Cross-backbone generalization of StructGEO. StructGEO achieves the best average performance for every optimizer backbone. Across cross-backbone settings, StructGEO achieves an

Table 3: Ablation study of the merge stages using `GPT-5 mini` as both the optimizer and the generative engine. *Merge* denotes the merge stages applied after leaf rewriting.

Merge		GEO-Bench			E-Commerce		
Section	**Document**	Pos	Word	**Overall**	Pos	Word	**Overall**
Original		19.85	19.91	19.96	18.07	17.93	18.04
Reorganization-only		23.05	22.96	23.04	21.35	20.96	21.15
✓	✓	**38.35**	**39.44**	**38.72**	**36.88**	**38.19**	**37.40**
✓	×	35.60	36.93	36.14	32.69	33.41	32.87
×	✓	31.92	33.68	32.39	31.61	32.52	31.94
×	×	26.13	27.31	26.61	25.33	25.92	25.59

average Word & Pos score of 33.82, outperforming IF-GEO (29.85) and AutoGEO (29.19) by 13.3% and 15.9%, respectively. These results show that StructGEO remains effective across heterogeneous optimizer and generative-engine backbones, demonstrating robust cross-backbone generalization.

6.3 Ablation Study

Table 3 examines how document reorganization and two merge stages contribute to StructGEO. *Reorganization-only* serves as a controlled structural baseline: it applies structural reorganization alone, without any intent modeling. When a merge stage is disabled, its input passages are simply concatenated, with no additional rewriting or reorganization.

Interestingly, reorganization alone is already effective. Reorganization-only improves Word & Pos by 15.4% and 17.2% over the original documents on GEO-Bench and E-Commerce, respectively, while changing document length only marginally. This suggests that visibility can be improved by restructuring content, not just by adding it. Applying leaf rewriting on top of Reorganization-only, still without either merge stage, adds further gains of 15.5% and 21.0% on GEO-Bench and E-Commerce, respectively, confirming the benefit of intent-specific rewriting.

Both merge stages contribute additional gains, and StructGEO with both stages performs best on every metric for both datasets. However, the two stages are not equally important. Starting from section merge alone, adding document merge yields gains of 7.1% and 13.8% on GEO-Bench and E-Commerce, respectively; starting from document merge alone, adding section merge yields larger gains of 19.5% and 17.1%. Consistently, when only one merge stage is used, section merge outperforms document merge by 11.6% and 2.9% on the two datasets. We attribute this to their different roles: section merge consolidates evidence for similar search intents at the section level, whereas document merge integrates the resulting sections into a coherent final document, providing complementary gains.

Finally, we further analyze query concentration in Appendix D. It shows that increasing leaf-level queries on a few dominant parent intents, while keeping their total number fixed, consistently reduces visibility, further underscoring the importance of broad intent coverage.

7 Conclusion

In this work, we proposed StructGEO, a GEO framework that addresses two questions together: *which search intents a document should be rewritten to serve*, and *how the evidence for those intents should be organized*. First, StructGEO builds a top-down intent tree that covers a broader range of search intents relevant to the target document. Second, it performs structure-aware rewriting that progressively groups evidence for related intents into coherent sections, keeping relevant evidence together rather than scattered across the document. Extensive experiments show that StructGEO consistently improves document visibility over existing GEO methods across multiple LLM-based generative engines. These results highlight that effective document optimization for generative engines depends not only on what information a document covers but also on how that information is organized.

AI USE STATEMENT

We used generative AI tools to polish the manuscript's language and to write and revise code. We supervised and verified all AI-assisted work and all reported results, and take responsibility for the final manuscript and artifacts.

REPRODUCIBILITY STATEMENT

The main text and appendix document the datasets, prompts, hyperparameter settings, procedures, and evaluation protocols used in our experiments. We plan to release the implementation code, and experiment configurations upon acceptance.

REFERENCES

Pranjal Aggarwal, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, Karthik Narasimhan, and Ameet Deshpande. GEO: generative engine optimization. In *KDD*, pp. 5–16, 2024.

Daria Alexander, Wojciech Kusa, and Arjen P. de Vries. ORCAS-I: queries annotated with intent using weak supervision. In *SIGIR*, pp. 3057–3066, 2022.

Puneet S. Bagga, Vivek F. Farias, Tamar Korkotashvili, Tianyi Peng, and Yuhang Wu. E-GEO: A testbed for generative engine optimization in e-commerce. *CoRR*, abs/2511.20867, 2025.

Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Networks*, 30(1-7):107–117, 1998.

Tong Chen, Jiawei Guo, Yuxi Li, Baiming Chen, Houxing Ren, Zhang Zhiwei, Yunxiang Zhang, Hanyang Xia, Kun Liang, and Zhaoran Fan. Mind reader: Latent user demand-guided content optimization for generative search engine. In *ACL*, pp. 40832–40848, 2026.

Xiaolu Chen, Haojie Wu, Jie Bao, Zhen Chen, Yong Liao, and Hu Huang. Role-augmented intent-driven generative search engine optimization. *CoRR*, abs/2508.11158, 2025.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Jimmy Lin. MS MARCO: benchmarking ranking models in the large-data regime. In *SIGIR*, pp. 1566–1576, 2021.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Retrieval augmented language model pre-training. In *ICML*, pp. 3929–3938, 2020.

Junqing He, Kunhao Pan, Xiaoqun Dong, Zhuoyang Song, YiBo Liu, Qianguo Sun, Yuxin Liang, Hao Wang, Enming Zhang, and Jiaxing Zhang. Never lost in the middle: Mastering long-context question answering with position-agnostic decompositional training. In *ACL*, pp. 13628–13642, 2024.

Cheng-Yu Hsieh, Yung-Sung Chuang, Chun-Liang Li, Zifeng Wang, Long T. Le, Abhishek Kumar, James R. Glass, Alexander Ratner, Chen-Yu Lee, Ranjay Krishna, and Tomas Pfister. Found in the middle: Calibrating positional attention bias improves long context utilization. In *ACL*, pp. 14982–14995, 2024.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *EACL*, pp. 874–880, 2021.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *EMNLP*, pp. 7969–7992, 2023.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. In *COLM*, 2025.

Sunghwan Kim, Wooseok Jeong, Serin Kim, Sangam Lee, and Dongha Lee. SAGEO arena: A realistic environment for evaluating search-augmented generative engine optimization. In *KDD*, pp. 2342–2353, 2026.

Elisabeth Kirsten, Jost Große Perdekamp, Qinyuan Wu, Mihir Upadhyay, Krishna P. Gummadi, and Muhammad Bilal Zafar. Characterizing web search in the age of generative AI. In *ACL*, pp. 10827–10848, 2026.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466, 2019.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *SOSP*, pp. 611–626, 2023.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *NeurIPS*, 2020.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In *EMNLP*, pp. 5420–5438, 2025.

Nelson F. Liu, Tianyi Zhang, and Percy Liang. Evaluating verifiability in generative search engines. In *EMNLP*, pp. 7001–7025, 2023.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Trans. Assoc. Comput. Linguistics*, 12:157–173, 2024.

Tie-Yan Liu. Learning to rank for information retrieval. In *SIGIR*, pp. 904, 2010.

Zikang Liu and Peilan Xu. Think before writing: Feature-level multi-objective optimization for generative citation visibility. In *ACL*, pp. 20290–20303, 2026.

OpenAI. Openai GPT-5 system card. *CoRR*, abs/2601.03267, 2026.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. LLM evaluators recognize and favor their own generations. In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

Haritz Puerto, Martin Gubri, Tommaso Green, Seong Joon Oh, and Sangdoo Yun. C-SEO bench: Does conversational SEO work? In *NeurIPS*, 2025.

Evangelia Spiliopoulou, Riccardo Fogliato, Hanna Burnsky, Tamer Soliman, Jie Ma, Graham Horwood, and Miguel Ballesteros. Play favorites: A statistical method to measure self-bias in llm-as-a-judge. *CoRR*, abs/2508.06709, 2025.

Runchu Tian, Yanghao Li, Yuepeng Fu, Siyang Deng, Qinyu Luo, Cheng Qian, Shuo Wang, Xin Cong, Zhong Zhang, Yesai Wu, Yankai Lin, Huadong Wang, and Xiaojiang Liu. Distance between relevant information pieces causes bias in long-context llms. In *ACL*, pp. 521–533, 2025.

Meiyun Wang, Takeshi Kojima, Yusuke Iwasawa, and Yutaka Matsuo. Lost in the distance: Large language models struggle to capture long-distance relational knowledge. In *NAACL*, pp. 4536–4544, 2025.

Beining Wu, Fuyou Mao, Jiong Lin, Cheng Yang, Jiaxuan Lu, Yifu Guo, Siyu Zhang, Yifan Wu, Ying Huang, and Fu Li. From experience to skill: Multi-agent generative engine optimization via reusable strategy learning. In *ACL*, pp. 43305–43315, 2026a.

Yujiang Wu, Shanshan Zhong, Yubin Kim, and Chenyan Xiong. What generative search engines like and how to optimize web content cooperatively. 2026b.

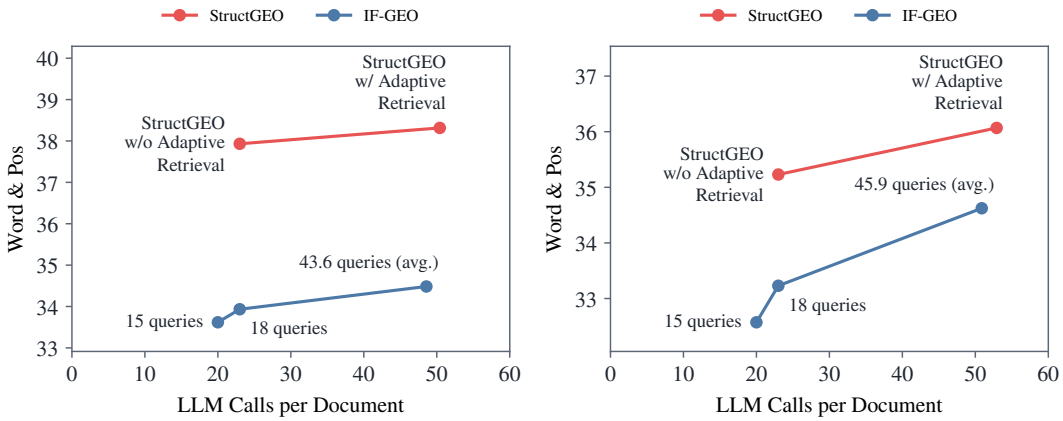
Junwei Yu, Mufeng Yang, Yepeng Ding, and Hiroyuki Sato. Structural feature engineering for generative engine optimization: How content structure shapes citation behavior. *CoRR*, abs/2603.29979, 2026.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *CoRR*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P. Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. Lmsys-chat-1m: A large-scale real-world LLM conversation dataset. In *ICLR*, 2024.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: less is more for alignment. In *NeurIPS*, 2023.

Heyang Zhou, Jiajia Chen, Xiaolu Chen, Jie Bao, Zhen Chen, and Yong Liao. IF-GEO: conflict-aware instruction fusion for multi-query generative engine optimization. In *ACL*, pp. 27576–27590, 2026.



(a) GEO-Bench (b) E-Commerce

Figure 4: Performance comparison between StructGEO and IF-GEO under matched LLM-call budgets on GEO-Bench and E-Commerce.

Appendix organization. The appendix provides additional analyses and implementation details that support the main experiments. Appendix A analyzes the computational complexity and LLM-call efficiency of StructGEO, and Appendix B evaluates the quality of GEO-rewritten documents. Appendices C, D, and E examine the effects of document length, query concentration, and the number of leaf intents, respectively. Appendix F presents qualitative examples showing how StructGEO reorganizes document content. Appendix G describes document segmentation, hyperparameter settings, and considerations for fair baseline comparison. Appendix H discusses the limitations of our work. Finally, Appendix I gives the prompt used for the Reorganization-only variant, and Appendix J gives the prompts and implementation details for each component of StructGEO.

A Complexity Analysis

Under the same query budget, both IF-GEO and StructGEO scale linearly with the number of modeled information needs. We compare their LLM-call complexity below.

Let Q denote the number of potential queries used by IF-GEO. For StructGEO, we consider a three-level intent tree, where N_{parent} is the number of parent intents, N_{leaf} the number of initially generated leaf intents, and $R \leq N_{\text{leaf}}$ the number of leaves with sufficient grounded evidence. Let $A \leq R$ and $B \leq A$ denote the numbers of adaptive-retrieval diagnosis and refinement calls, respectively.

IF-GEO requires one call per potential query, followed by five additional calls for query generation, deduplication, conflict resolution, blueprint generation, and final rewriting. Thus, $T_{\text{IF-GEO}}(Q) = Q + 5 = \Theta(Q)$.

For StructGEO, intent-tree construction requires $1+N_{\text{parent}}$ calls: one to generate broad search intents and N_{parent} to derive leaf search intents from their respective parents. Structure-aware rewriting requires $R + A + B + N_{\text{parent}} + 1$ calls, including N_{parent} section merges and one document merge. Thus, $T_{\text{StructGEO}} = R + A + B + 2N_{\text{parent}} + 2$. For a complete three-level tree with N_{parent} parent intents and C children per parent, assuming all $N_{\text{parent}}C$ leaves are retained and both adaptive-retrieval calls are executed for every leaf,

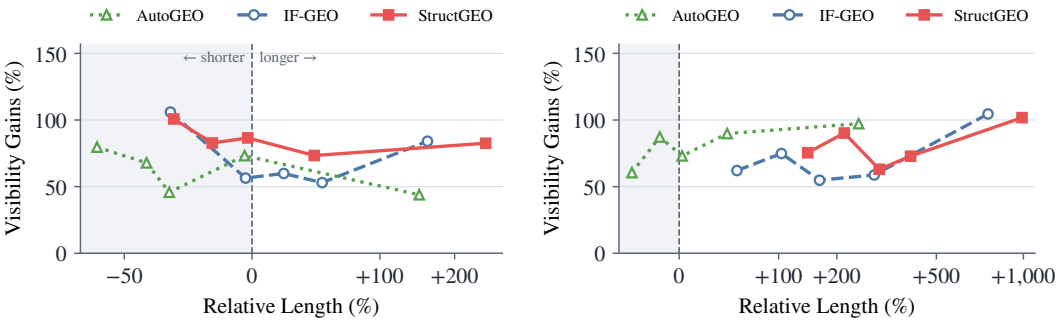
$$T_{\text{StructGEO}}(N_{\text{parent}}, C) \leq 3N_{\text{parent}}C + 2N_{\text{parent}} + 2 = O(N_{\text{parent}}C).$$

More generally, its complexity is $O(N_{\text{leaf}})$. Under our experimental setting, where $N_{\text{leaf}} = Q$, both StructGEO and IF-GEO have the same asymptotic complexity of $O(Q)$.

To further examine whether StructGEO's performance gains can be attributed simply to a larger number of LLM calls, we conduct a call-matched comparison with IF-GEO. As shown in Figure 4, we increase the number of queries used by IF-GEO to match the LLM-call budgets of StructGEO with and without adaptive retrieval. At approximately 23 calls per document, StructGEO without adaptive retrieval substantially outperforms IF-GEO with 18 queries on both GEO-Bench and E-Commerce.

Table 4: Quality of the target document, evaluated with GPT-4o-mini. The best performance is shown in **bold**, and the second-best is underlined.

Backbone	Metric	GEO-Bench				E-Commerce			
		Vanilla	AutoGEO	IF-GEO	**StructGEO**	Vanilla	AutoGEO	IF-GEO	**StructGEO**
GPT-5 mini	Content↑	77.02	82.50	82.17	**84.96**	73.45	82.14	79.66	**82.54**
	Appeal↑	70.27	74.30	75.53	**78.43**	65.93	74.26	72.05	**75.27**
	Qual↑	74.32	79.22	79.51	**82.35**	70.44	78.99	76.62	**79.63**
Gemini-3.1 Flash-Lite	Content↑	76.80	82.08	81.76	**83.01**	74.10	79.90	78.47	**80.36**
	Appeal↑	69.78	74.72	74.55	**75.34**	66.50	71.56	70.68	**72.69**
	Qual↑	73.99	79.14	78.88	**79.94**	71.06	76.56	75.35	**77.29**
Qwen3.8-27B	Content↑	76.90	82.65	82.27	**85.60**	73.63	82.08	79.04	**82.53**
	Appeal↑	69.84	74.44	74.52	**78.14**	65.00	72.58	69.97	**74.22**
	Qual↑	74.08	79.37	79.17	**82.61**	70.18	78.28	75.41	**79.21**



(a) Gemini-3.1 Flash-Lite

(b) GPT-5 mini

Figure 5: Performance over relative document length on E-Commerce. We partition instances into five groups according to the percentage change in token count from the original to the rewritten document and use each group's median relative length as its x-value. Word & Pos gain is computed as the improvement in the group-level mean Word & Pos score over the corresponding mean Vanilla score.

We observe the same trend at the larger call budget. We increase IF-GEO to an average of 43.6 queries on GEO-Bench and 45.9 on E-Commerce so that its call budget approximately matches StructGEO with adaptive retrieval, and StructGEO still achieves higher Word & Pos scores. These results suggest that StructGEO's gains cannot be explained solely by a larger LLM-call budget and are consistent with the benefits of hierarchical intent modeling and structure-aware rewriting.

B Quality of GEO-Rewritten Documents

We evaluate the quality of GEO-rewritten target documents using the Quality (Qual) metric from FeatGEO (Liu & Xu, 2026). An LLM judge evaluates four content aspects (*i.e.*, fluency, usefulness, credibility, and structure) and three appeal aspects (*i.e.*, uniqueness, attractiveness, and influence on the generated response). The former are averaged into Content and the latter into Appeal, and the overall quality score is computed as $\mathrm{Qual} = 0.6\,\mathrm{Content} + 0.4\,\mathrm{Appeal}$.

Table 4 shows that StructGEO achieves the highest Content, Appeal, and Qual scores across all six settings. Compared with Vanilla, StructGEO improves Content, Appeal, and Qual by up to 12.38%, 14.18%, and 13.05%, respectively. It also outperforms the strongest competing GEO method by up to 3.57%, 4.86%, and 4.08%, respectively. These results indicate that StructGEO improves visibility without sacrificing target-document quality; rather, its visibility gains are accompanied by higher-quality and more appealing target documents.

Table 5 further breaks down the seven underlying dimensions. StructGEO consistently improves the individual content dimensions, with particularly strong gains in credibility, usefulness, and structure,

Table 5: The seven FeatGEO Qual dimensions of the target document. Each 1–5 rating is normalized to $[0, 1]$ as $(s-1)/4$ and reported $\times 100$. The best score is marked in **bold** and the second-best is underlined.

Backbone	Method	Content				Appeal		
		Fluency	Usefulness	Credibility	Structure	Uniqueness	Attractiveness	Influence
GEO-Bench								
GPT-5 mini	Vanilla	75.85	79.88	74.17	78.18	63.48	71.41	75.93
	AutoGEO	80.94	84.58	79.87	84.62	68.33	75.22	79.34
	IF-GEO	79.85	84.58	80.78	83.48	69.87	75.83	80.88
	StructGEO	**81.78**	**86.90**	**85.18**	**85.99**	**73.04**	**78.45**	**83.80**
Gemini-3.1 Flash-Lite	Vanilla	75.60	79.46	73.87	78.28	62.64	71.14	75.57
	AutoGEO	80.36	83.82	79.20	84.92	69.47	75.61	79.09
	IF-GEO	80.46	84.58	78.38	83.63	68.42	75.59	79.65
	StructGEO	**81.57**	**84.61**	**80.45**	**85.42**	**69.86**	**76.24**	**79.92**
Qwen3.8-27B	Vanilla	75.80	79.68	73.96	78.18	62.67	71.16	75.70
	AutoGEO	81.57	84.27	79.64	85.11	68.62	75.52	79.20
	IF-GEO	81.20	84.15	79.47	84.24	68.33	75.47	79.76
	StructGEO	**83.11**	**86.77**	**85.54**	**86.96**	**73.06**	**78.36**	**82.99**
Average	Vanilla	75.75	79.68	74.00	78.21	62.93	71.23	75.73
	AutoGEO	80.96	84.22	79.57	84.88	68.81	75.45	79.21
	IF-GEO	80.50	84.43	79.54	83.78	68.88	75.63	80.09
	StructGEO	**82.15**	**86.09**	**83.72**	**86.12**	**71.99**	**77.68**	**82.24**
E-Commerce								
GPT-5 mini	Vanilla	73.79	71.48	71.88	76.65	58.47	69.08	70.25
	AutoGEO	**82.89**	**81.93**	77.69	**86.04**	67.84	76.61	78.32
	IF-GEO	79.54	78.12	77.60	83.37	65.78	74.57	75.80
	StructGEO	81.07	81.59	**81.84**	85.65	**69.59**	**76.70**	**79.53**
Gemini-3.1 Flash-Lite	Vanilla	73.87	72.97	72.25	77.32	58.69	69.36	71.44
	AutoGEO	**81.33**	76.03	77.35	**84.89**	66.13	74.58	73.98
	IF-GEO	79.29	76.79	75.47	82.31	63.57	74.05	74.41
	StructGEO	81.19	**78.49**	**77.37**	84.40	**67.02**	**75.13**	**75.91**
Qwen3.8-27B	Vanilla	74.53	69.86	72.34	77.78	57.54	68.83	68.63
	AutoGEO	**84.18**	76.52	80.54	87.06	67.65	75.81	74.28
	IF-GEO	80.83	74.76	76.67	83.88	63.07	73.77	73.07
	StructGEO	83.03	**77.97**	**81.75**	**87.36**	**69.28**	**76.59**	**76.79**
Average	Vanilla	74.06	71.44	72.16	77.25	58.23	69.09	70.11
	AutoGEO	**82.80**	78.16	78.53	**86.00**	67.21	75.67	75.53
	IF-GEO	79.89	76.56	76.58	83.19	64.14	74.13	74.43
	StructGEO	81.77	**79.35**	**80.32**	85.80	**68.63**	**76.14**	**77.41**

while also improving all three appeal dimensions. These trends are consistent with the design of StructGEO: organizing information around the potential search intents and strengthening evidential support make the rewritten document more useful, credible, and structurally effective as a source for the generative engine.

C Effect of Document Length on Visibility

Figure 5 examines whether the visibility improvements of GEO methods can be explained solely by changes in document length. We compare AutoGEO, IF-GEO, and StructGEO on E-Commerce using Gemini-3.1 Flash-Lite and GPT-5 mini. We exclude instances with zero visibility in the original document and measure document length using the `o200k_base` tokenizer.

We observe that **visibility gain does not increase monotonically with document length**. Larger document expansion does not necessarily lead to higher visibility. In particular, StructGEO maintains visibility gains across a broad range of document-length changes, including cases where the rewritten document is shorter than the original. As shown in Figure 5(a), StructGEO achieves gains for both shorter and longer rewrites, with some shorter-document groups outperforming groups with substantially larger document expansions.

The two backbones also exhibit markedly different ranges of document-length changes. With Gemini-3.1 Flash-Lite, rewritten documents range from shorter than the original to moderately longer (Figure 5(a)), whereas GPT-5 mini generally exhibits substantially larger expansions (Figure 5(b)).

These results indicate that GEO performance improvements cannot be explained solely by document expansion. Together with our structural analysis, they suggest that factors beyond document length, including how information is rewritten and organized, contribute to document visibility.

D Effect of Query Concentration

We conduct a controlled query-bias experiment on E-Commerce using GPT-5 mini to examine how the concentration of potential queries affects StructGEO. Starting from a balanced 3×3 intent tree with three parent intents and nine leaf nodes in total, we keep the total number of leaf nodes fixed while varying their allocation across parents as $(3, 3, 3)$, $(5, 2, 2)$, $(7, 1, 1)$, and $(9, 0, 0)$. These configurations correspond to top-1 leaf concentrations of 33%, 56%, 78%, and 100%, respectively.

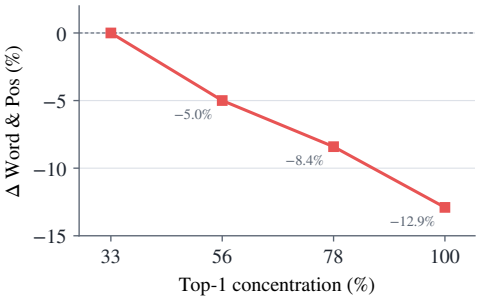


Figure 6: Effect of query concentration on StructGEO. With a fixed leaf-level query budget, visibility decreases as leaf nodes become increasingly concentrated on a single parent intent.

Figure 6 shows the effect of increasing query concentration while keeping the leaf-level query budget fixed. As the top-1 concentration increases, Word & Pos consistently decreases, with relative drops of 5.0%, 8.4%, and 12.9% at 56%, 78%, and 100% concentration, respectively. These results suggest that covering diverse aspects of a target document with potential queries is more effective than concentrating the same query budget on a single intent.

E Hyperparameter Sensitivity Analysis

We examine the effect of varying the number of leaf nodes per parent intent over $1, 3, 5, 7, 9$ while keeping the other hyperparameters fixed. As shown in Figure 7, Word & Pos visibility consistently increases as the number of leaf nodes per parent grows, from 34.4 with one leaf node per parent to 39.8 with nine leaf nodes per parent. Since each leaf node represents a fine-grained potential information need and directly induces an intent-specific rewrite, increasing the number of leaf nodes per parent allows StructGEO to capture a broader range of fine-grained information needs under each parent intent and incorporate more query-specific rewriting signals before the resulting content is hierarchically merged. This trend is consistent with the observation in IF-GEO that increasing the number of inferred queries improves overall visibility performance. However, increasing the number of leaf nodes per parent also raises the optimization budget by requiring more intent-specific rewrites. We therefore use five leaf nodes per parent as the default setting to maintain a practical optimization budget while retaining substantial visibility gains.

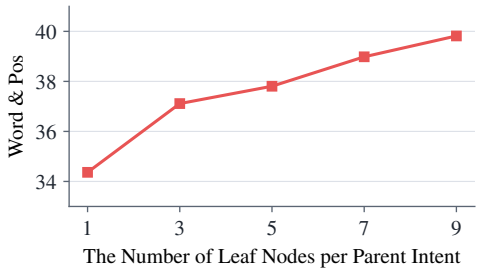


Figure 7: Performance across different numbers of leaf nodes in the intent tree on E-Commerce using Gemini-3.1 Flash-Lite.

F Qualitative Analysis

To examine how our method restructures a document, we conduct a qualitative analysis. The first row lists the broad intents extracted by StructGEO, and the sentences in the original and rewritten documents are colored according to the intent they serve. In the documents rewritten by AutoGEO and IF-GEO, sentences associated with the same broad intent remain scattered across the document. StructGEO, in contrast, first extracts top-level intents from the entire document, rewrites the evidence

Table 6: Case study on an E-commerce document (sample 1997). The first row shows the top-level intents extracted by StructGEO, and sentences in the original and rewritten documents are colored by the intent they serve. In the baseline rewrites, evidence associated with the same intent remains interleaved, whereas StructGEO restructures the whole document around the broad search intents.

Top-level intents extracted by StructGEO

Intent 1 What are the best walking shoes to wear in Paris?
Intent 2 What shoes should I pack for Paris by season?
Intent 3 What kinds of shoes do Parisians wear?

Original Document

Most people associate Paris with fashion. Ok, and the Eiffel Tower. [. . .] Flats and sneakers are the most common shoe choice in Paris. Simply white sneakers like this model from Lacoste are a perfectly solid choice. [. . .] This means sneakers are not only worn because they are comfortable, but also because they complete a look. [. . .] Sneakers are the best comfortable walking shoes for Paris and luckily totally appropriate! [. . .] I noticed that Veja is especially often worn by women over 35. [. . .] Generally, all sneakers that I mentioned above are also acceptable in the evening when going to casual bars or restaurants. However, more sporty shoes like running shoes are better just worn during the daytime when exploring Paris. [. . .] Brogues or Oxfords are great travel shoes — they can easily be dressed up and down, perfect for a day sightseeing but also for an evening in a fancy restaurant. They [. . .] are also a safe bet during shoulder seasons. [. . .] If I had to pick, I would say that classic ankle boots are the 2nd most favorite shoes of the Parisians — after sneakers. Especially in winter and during the shoulder seasons of spring and autumn, they are a great alternative to loafers. [. . .] If you travel to Paris during December [. . .] waterproof boots can help you a lot to keep your feet dry and warm [. . .] The ones from Blondo are my favorites, but the Chelsea boots from Sorel are nice as well.

Rewritten Document by AutoGEO

Source metadata: Author: Lena — last updated 19 April 2022. Top (direct) recommendation: Pack one pair of comfortable sneakers (white or neutral), one pair of versatile flats/loafers or brogues, and — if traveling in autumn/winter — one pair of waterproof ankle or Chelsea boots. [. . .] Veja: Typical price range: approx. €70–€140. Why: Popular in Paris for combining style and casual polish; good as a fashion-forward alternative to classic white trainers. Usage notes: Often favored by adults 30+; suitable for daytime and many casual evenings. Nike Air Max: Why: Comfortable for long walks; many Parisians wear Air Max for daytime exploring. [. . .] Brogues / Oxfords: Why: One of the most versatile travel shoes — works with jeans, slacks, and skirts; good for sightseeing and nicer dinners. [. . .] Waterproof ankle boots (Blondo, Sorel or similar): Why: Best for rainy Paris days in autumn/winter — keep feet dry and warm. Chelsea boots: Why: Timeless, streamlined and highly wearable in Paris; works across seasons except high summer. [. . .] Rain winter: Paris winters are often wet rather than snowy — waterproof boots or treated leather are more useful than snow boots. Evening wear: Fashion sneakers, loafers, brogues and Chelsea boots are acceptable for most casual-to-smart evenings. Breaking in: Always break in new shoes before travel to avoid blisters.

Rewritten Document by IF-GEO

How to choose walking shoes for Paris: Practical selection criteria — short, objective checklist to guide your purchases and product-page scans. Cushioning: Look for a cushioned midsole (foam, EVA, or visible air units) to reduce fatigue on all-day sightseeing. [. . .] Spring (late March–May): Recommended shoe types: lightweight ankle boots (water-resistant leather), comfortable sneakers, and loafers for drier days. Sneakers (long walks): New Balance 990 (mid — supportive with good cushioning), Nike Pegasus (mid — cushioned, breathable). [. . .] Winter (December–February): Winter priorities: waterproofing, insulation, and durable grip. Waterproof boots (long walks): Blondo waterproof ankle boots (mid — waterproof membrane; moderate support). [. . .] The Parisian everyday fashion is a combination of (intentional and well-placed) effortless, a lot of class and comfort. [. . .] Brand-function notes: Veja V-10 — stylish, eco-conscious silhouette with moderate cushioning; better for shorter days or pair with a supportive insole for long walks. Nike Air Max — very popular and roomy cushioning (best for long walks thanks to visible air units). [. . .] If I had to pick, I would say that classic ankle boots are the 2nd most favorite shoes of the Parisians — after sneakers. Especially in winter and during the shoulder seasons of spring and autumn, they are a great alternative to loafers.

Rewritten Document by StructGEO (ours)

Best Sneaker Choices (Daytime Walking)
Comfortable for long days, practical on mixed surfaces, and widely acceptable. For an all-day single pair, neutral leather-look sneakers (white or other neutrals) are smart [. . .] Styling for evenings: Context matters: athletic/technical trainers are best for long daytime walks. Clean minimalist leather or fashion sneakers (Veja-style) work for casual evenings (bistros, bars). [. . .]

Parisian Shoe Style — What Locals Wear
Flats and sneakers are most common. Simple white leather trainers are a staple. Sneakers are used deliberately to complete looks — to dress down tailored pieces or create effortless combinations. [. . .] Veja is notably popular among Parisians. Loafers, brogues and derbies are also widely worn — versatile for day and evening. Chelsea boots and ankle boots are staples for cooler or wet months. Leave high stilettos at home for most sightseeing.

Packing Shoes for Paris — By Season
Year-round: one pair of sneakers. Spring [. . .] loafers are a top choice [. . .] Summer: sandals + breathable sneakers [. . .] Autumn: ankle boots are ideal [. . .] Winter: Paris winters are typically wet and chilly rather than severely cold. Waterproof Chelsea or ankle boots [. . .] are practical. Two representative options: Blondo waterproof ankle boot [. . .] Sorel waterproof Chelsea [. . .]

for each fine-grained intent level, and then merges the resulting content upward under similar search intents. This global restructuring is visible in Table 6, where each StructGEO section is dominated by a single intent color.

G Reproducibility

G.1 Document Segmentation

We segment each source document into evidence units, referred to as segments, using a rule-based procedure. We first normalize line breaks and split the document into blocks at blank lines; if no blank lines are present, we use single line breaks as block boundaries, and if no line breaks are present, we treat the entire document as a single block. Blocks of at most 80 words are retained as individual segments, whereas longer blocks are further split into sentences. Sentence boundaries are identified at whitespace following sentence-final punctuation when the next character is an uppercase letter, digit, quotation mark, or opening parenthesis. If this process produces only a single segment, we re-segment the entire document using the same sentence-boundary rules. Finally, we remove duplicate segments after whitespace normalization, retaining only the first occurrence.

G.2 Detailed Hyperparameter Settings

Unless otherwise specified, we fix the number of parent intents to three and generate up to five child intents for each parent, yielding a maximum default budget of 15 leaf queries. Across all GEO methods, including StructGEO, we use a rewriting temperature of 0.2 whenever supported by the backbone model. For answer generation by the target generative engine during final evaluation, we set the temperature to 0.0 for all methods. We serve Qwen3.8-27B with FP8 weights using vLLM (Kwon et al., 2023) on an H200 GPU and use Qwen3-Embedding-8B for retrieval. For the M1 motivating analysis in Section 3, we use GPT-5 mini to generate IF-GEO's potential queries and identify the broad search intents of each target document, and GPT-5.6 Luna to assign each generated query to one of the identified intents. Except in the cross-backbone experiment, we use the same backbone model for both the optimizer and the generative engine. For all remaining model-specific hyperparameters, we follow the settings reported in the corresponding papers.

As an additional observation, evidence-threshold pruning is rarely triggered. After grounding, only 9 trees (0.07%) lose any intent. Thus, with $\tau = 0.20$, the threshold primarily serves as a safeguard for removing unsupported intents rather than substantially reducing the intent-tree size.

G.3 Considerations for Baseline Comparison

We exclude FeatGEO (Liu & Xu, 2026) and Mind Reader (Chen et al., 2026) from our comparison because they require additional query- or engine-specific information that is unavailable under the black-box optimization setting adopted in most prior GEO work (Aggarwal et al., 2024; Chen et al., 2025; Bagga et al., 2025; Wu et al., 2026b; Zhou et al., 2026). Specifically, most prior GEO work considers a black-box setting in which the internal mechanisms of the generative engine are inaccessible and the user queries associated with a target document are unobservable at optimization time. FeatGEO relies on topic-specific query probes and citation feedback from the generative engine, whereas Mind Reader requires the ground-truth user query and query-specific reasoning information for optimization. Providing such information would violate this setting and lead to an unfair comparison.

We also exclude GEO-SFE (Yu et al., 2026) from our comparison because it cannot be faithfully reproduced from the paper alone. Although the paper describes the overall optimization framework and several structural principles, key implementation details are not fully specified, including the training and configuration of the citation predictor, the construction of target structural features, and the thresholds and operators used for structural transformations. In the absence of these details or an official implementation, reproducing GEO-SFE would require substantial implementation-specific assumptions, making a direct comparison potentially unreliable.

H Discussion and Limitations

Broader impact and potential misuse. While GEO can help useful information become more visible in generative-engine responses, it may also enable *visibility gaming*, where documents are optimized to attract citations independently of their underlying information quality. This risk is particularly important because our objective optimizes visibility rather than truthfulness, and higher

visibility should therefore not be interpreted as greater reliability or authority. StructGEO does not provide a general safeguard against misleading or strategically crafted source content. Future work should consider GEO objectives that jointly account for visibility, source faithfulness, factual reliability, and resistance to manipulation.

End-to-end GEO. Recent work (Kim et al., 2026) has highlighted a limitation of conventional GEO evaluation and introduced an end-to-end pipeline that jointly considers retrieval, reranking, and generation. Our evaluation focuses on document visibility after the target document has already been included in the candidate set. Following prior GEO work (Aggarwal et al., 2024; Chen et al., 2025; Bagga et al., 2025; Wu et al., 2026b; Zhou et al., 2026; Liu & Xu, 2026; Chen et al., 2026), we assume a fixed candidate set and do not model upstream retrieval or ranking mechanisms, as the retrieval systems used by real-world generative engines are, to the best of our knowledge, not publicly disclosed. Therefore, StructGEO optimizes visibility conditional on retrieval rather than addressing the full end-to-end GEO pipeline from retrieval to generation. Extending StructGEO to jointly consider retrieval-stage inclusion and generation-stage visibility remains an important direction for future work.

Adaptive intent tree construction. StructGEO currently uses a fixed number of parent and child intents across documents. However, the optimal number and granularity of search intents may vary depending on the content and complexity of each target document. An adaptive intent-tree construction strategy could therefore dynamically determine the number of intents and tree depth on a per-document basis, allowing StructGEO to allocate its query budget more efficiently. We leave this promising direction for future work.

I Prompts Used for the Reorganization-only Variant

System Prompt

```
You are a source-faithful document editor. Reorganize only the
information explicitly present in the supplied document into a clear
and coherent structure.
```

User Prompt

```
Task:
Directly restructure the complete target document into clear, self-
contained Markdown.

Experimental constraint:
- Derive the organization only from topics and relationships
explicitly visible in the target document.

Grounding and preservation rules:
- Use only information explicitly present in the target document.
- Do not add external facts, unsupported claims, invented examples,
citations, numbers, product attributes, or recommendations.
- Preserve all meaningful information in the source document,
including every distinct fact, step, number, name, condition,
limitation, comparison, qualification, and product detail exactly.

Structuring rules:
- Identify topically and semantically related information across the
entire document, even when it appears in distant passages, and place
it together in the same coherent section.
- Organize each section around one clear topic or concept under a
descriptive Markdown heading.
```

```
- Avoid scattering information about the same topic across multiple
sections and avoid repeating the same source information in different
sections.
- Do not merge distinct facts, conditions, perspectives, entities, or
time periods merely because they share superficial keywords; preserve
meaningful distinctions within subsections or separate sections.
- Use paragraphs, bullet lists, numbered steps, or tables only when
the source content supports that form.
- Place definitions, conclusions, conditions, and supporting details
in a logical order.
- Remove only duplicated presentation artifacts or obvious navigation
noise; do not remove substantive content.
- Return the entire restructured document, not a plan, explanation,
summary, or analysis.

Output format:
- Return only the final restructured document.

Target Document:
{document}
```

J Implementation Details of StructGEO Components

J.1 Intent Tree Construction

J.1.1 Parent Intent Generation

System Prompt

```
You are an expert in identifying potential user information needs from
 webpages. Your task is to identify the broad information needs that
explain why users may search for a given webpage and that can be
further refined into narrower child intents. Infer the broad search
intents that explain why users would search for a given webpage.
```

User Prompt

```
Task Description:
Predict the broad parent search intents for the given webpage before
any child intents are generated.

Requirements:
- Generate exactly <PARENT_COUNT> parent intents unless fewer are
genuinely supported.
- Each parent intent must be broad enough to contain several narrower
child intents.
- Parent intents must be inferred only from the target webpage text.
- Parent intents should capture the main information needs that would
motivate users to search for the webpage.
- Do not generate child intents in this step.
- Use concise natural-language search-query phrasing.
- Infer a concise snake_case intent_type label for each intent.

Output Format:
Return ONLY a valid JSON object in the following structure:
{
  parent_intents: [
    {
      intent_query: <broad parent search intent>,
```

```
      intent_label: <short label>,
      intent_type: <model_inferred_label>
    }
  ]
}

Target Webpage Text:
<TARGET_WEBPAGE_TEXT>
```

J.1.2 CHILD INTENT GENERATION

System Prompt

```
You are an expert in identifying potential user information needs from
 webpages. Given a single parent search intent, infer child intents
that are strictly narrower and semantically dependent on the parent.
```

User Prompt

```
Task Description:
Given a parent search intent, generate child search intents that are
narrower and semantically dependent on that parent.

Parent Intent:
- query: <PARENT_QUERY>
- label: <PARENT_LABEL>
- type: <PARENT_INTENT_TYPE>

Requirements:
- Generate <CHILD_COUNT> child intents.
- Every child intent must be narrower than the parent and dependent on
 the parent.
- Infer child intents solely from the target webpage text and the
supplied parent intent.
- Use concise natural-language search-query phrasing.
- Ensure that the child intents are mutually distinct and avoid
substantial semantic overlap.
- Express each intent_query as a concise, natural-language search
query.
- Infer a concise snake_case intent_type label for each intent.

Output Format:
Return ONLY a valid JSON object in the following structure:
{
  child_intents: [
    {
      intent_query: <narrow child search intent>,
      intent_label: <short label>,
      intent_type: <model_inferred_label>
    }
  ]
}

Target Webpage Text:
<TARGET_WEBPAGE_TEXT>
```

J.2 LEAF REWRITE

J.2.1 INITIAL REWRITE

System Prompt

```
You are an expert Content Engineer and GEO (Generative Engine
Optimization) Specialist. First diagnose how the source passage falls
short of answering the leaf intent and convert each shortfall into a
rewriting rule, then apply every rule to produce one self-contained
rewritten leaf passage. Return exactly the two tagged sections
requested.
```

User Prompt

```
Task:
- Evaluate how well the source passage, as written, answers the leaf
intent for a reader with no other source.
- Convert each concrete shortfall into a rewriting rule.
- Rewrite the source passage into a complete, self-contained rewritten
 passage by applying every rule.

Rule Generation (inside REWRITING_RULES):
- Diagnose up to 5 shortfalls in the source passage relative to the
leaf intent, such as:
(1) information requested by the intent that is only briefly stated in
 the source passage;
(2) a fact given without its explanation, condition, or implication;
(3) key information buried in unrelated text;
(4) vague or ambiguous wording;
(5) content whose organization could be improved with headings, lists,
 or tables.
- For each shortfall, write one actionable rule specifying:
(1) the part of the source passage it targets,
(2) what should be added, clarified, reorganized, or simplified,
(3) the content that should be expressed to better address the leaf
intent.

Rewriting (inside REWRITTEN_PASSAGE):
- Apply every generated rule. Each rule is mandatory.
- Preserve all distinct information in the source passage, including
every distinct fact, step, number, name, condition, limitation,
comparison, qualification, and product detail exactly.
- Following the rules, clarify, restructure, and elaborate on the
source passage, but do not omit, shorten, or replace distinct
information with vague summaries.
- Develop each point requested by the intent in depth: state what it
is, why it matters, and under what conditions it applies.
- Use standard Markdown and apply any headings, lists, and tables
specified by the rules.
- Do not mention the leaf intent, the rules, or any internal process
in the rewritten passage.

Output Contract:
- Return exactly two tagged sections in the order below and nothing
else.
- In REWRITING_RULES, output exactly one rule per line, each beginning
 with - .
- In REWRITTEN_PASSAGE, output only the complete rewritten passage in
Markdown.
- Always close both tags.
```

```
- Do not wrap either section in a code fence.
- Do not return JSON.

Leaf Intent:
{leaf_intent}

Source Passage:
{aligned_evidence}

Required Output Format:
<REWRITING_RULES>
- <rule>
</REWRITING_RULES>

<REWRITTEN_PASSAGE>
<complete rewritten Markdown passage>
</REWRITTEN_PASSAGE>
```

J.2.2 Diagnosis of Missing Information and Corresponding Query Generation

System Prompt

```
Analyze the leaf intent, draft, and evidence. Generate source-
retrieval queries for useful missing information. Return JSON only.
```

User Prompt

```
Task:
Compare the draft against its leaf intent and evidence, and identify
specific information needs that would enable the draft to address the
leaf intent more concretely.

Rules:
- Use the leaf intent, draft, and leaf evidence to identify
information that is missing or insufficiently explained in the draft.
- For each useful information need, write a standalone search query
for the original webpage.
- Do not request information that the draft already explains
adequately.
- Return an empty support_points list if no missing information need
is apparent.
- Do not request generic length increases or formatting-only changes.

Output Format (JSON Only):
Return JSON:
{
  support_points: [
    {
      missing_information: <missing_information>,
      reason: <brief reason>,
      retrieval_query: <retrieval_query>
    }
  ]
}

Leaf Support Inputs:
{
  leaf_intent: <LEAF_INTENT_QUERY>,
  draft: <SAVED_INITIAL_LEAF_DRAFT>,
```

```
  leaf_evidence: [<EVIDENCE_TEXT>, ...]
}
```

J.2.3 PASSAGE REFINEMENT

System Prompt

```
You are an expert Content Engineer and GEO (Generative Engine
Optimization) Specialist.
```

User Prompt

```
Task:
Revise the draft written for the leaf intent using the missing
information needs and their supporting evidence.

Goals:
- Retain useful existing material and develop the explanation where
the additional context helps.
- Improve completeness, specificity, and flow.

Rules:
- Continue normal explanatory writing.
- The information needs are not a required checklist.

Output:
Return only the complete revised Markdown passage, without diagnostic
commentary.

Leaf Support Inputs:
{
  leaf_intent: <LEAF_INTENT_QUERY>,
  draft: <SAVED_INITIAL_LEAF_DRAFT>,
  leaf_evidence: [<EVIDENCE_TEXT>, ...],
  support_pairs: [
    {
      missing_information: <MISSING_INFORMATION>,
      brief reason: <REASON>,
      supporting_evidence: [<ADDITIONAL_EVIDENCE_TEXT>, ...]
    },
    ...
  ]
}
```

J.3 SECTION MERGE

The following system prompt is shared by the parent and root merge calls.

System Prompt

```
You are an expert Content Engineer and GEO (Generative Engine
Optimization) Specialist. Your task is to merge rewritten section
drafts according to a webpage-derived query-intent hierarchy, while
preserving factual accuracy and the distinct information contributed
by each passage.
```

User Prompt

```
Task:
Merge the rewritten child passages into one coherent rewritten section
 for the parent intent below.

Parent Intent:
- label: <PARENT_INTENT_LABEL>
- predicted_query_intent: <PARENT_INTENT_QUERY>
- intent_type: <PARENT_INTENT_TYPE>

Rules:
- Organize the merged content around the parent intent, keeping
relevant child-level details together in a coherent structure.
- Preserve every distinct fact, step, number, attribute, name,
constraint, comparison, and product detail present in the rewritten
child passages, including different information about the same topic.
Consolidate repeated statements of the same claim by merging their
wording, without replacing distinct details with a summary.
- Do not mention the merge process, intent hierarchy, evidence
identifiers, or any internal metadata.
- Return only Markdown content.

Rewritten Child Passages:
[
  {
    intent_label: <CHILD_INTENT_LABEL>,
    predicted_query_intent: <CHILD_INTENT_QUERY>,
    text: <REWRITTEN_CHILD_PASSAGE>
  }
]

Source Evidence Directly Aligned to This Parent:
<PARENT_ALIGNED_EVIDENCE>
```

J.4 DOCUMENT MERGE

System Prompt

```
You are an expert Content Engineer and GEO (Generative Engine
Optimization) Specialist. Your task is to merge rewritten section
drafts according to a webpage-derived query-intent hierarchy, while
preserving factual accuracy and the distinct information contributed
by each passage.
```

User Prompt

```
Task:
Synthesize the final rewritten webpage from a set of top-level section
 drafts.
Each section draft represents a major webpage-derived information need
 and incorporates its supporting details.

Goal:
Produce a complete and coherent webpage in Markdown. The final output
should read naturally as a webpage rather than as a collection of
independently generated passages.

Rules:
```

```
- Use the top-level section drafts as the primary structure of the
final webpage.
- Preserve each top-level draft as a distinct major section and keep
its details within that section.
- Maintain the semantic boundaries between sections while improving
transitions and overall webpage flow.
- Use the root evidence to preserve webpage-level context and recover
relevant information that may not be fully represented in the section
drafts.
- Integrate useful unassigned source evidence into the most relevant
existing section.
- If useful unassigned evidence does not naturally fit any existing
section, place it in an Additional Details section.
- Preserve every distinct fact, step, number, attribute, name,
constraint, comparison, and product detail introduced by the child
passages, including different information about the same topic.
Consolidate repeated statements of the same claim by merging their
wording, without replacing distinct details with a summary.
- Do not mention the merge process, intent hierarchy, evidence
identifiers, or any internal metadata.
- Return only the complete final webpage in Markdown. Do not output
analysis, JSON, comments, diagnostic text, or explanations.

Top-level Section Drafts:
[
  {
    intent_label: <PARENT_INTENT_LABEL>,
    predicted_query_intent: <PARENT_INTENT_QUERY>,
    text: <MERGED_PARENT_SECTION>
  }
]

Root Evidence Retrieved by Full-webpage Embedding:
<ROOT_ALIGNED_EVIDENCE>

Unassigned Source Evidence:
<UNASSIGNED_EVIDENCE>
```