STATEFUL ITERATIVE PARSING FOR GLOBAL-COHERENT HIERARCHICAL STRUCTURE EXTRACTION FROM UNBOUNDED DOCUMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Parsing long, unstructured documents into hierarchical machine-readable formats faces a fundamental tension between document length and limited LLM context. Prevailing one-shot or stateless chunk-based methods suffer from the "lost-in-the-middle" phenomenon and fragmented structures. We propose the **Iterative Structured Content Filling (ISCF)** framework, which incrementally builds a global tree through LLM-driven steps under partial observability. ISCF integrates three pillars: (1) **Dynamic Boundary Re-alignment (DBR)** for unsupervised chunk optimization via semantic coherence and uncertainty; (2) **Adaptive Structural Masking & Hierarchical-Aware Path Embedding (HAPE)** for efficient global awareness within finite context; (3) **Unique Content-ID Mapping** ensuring near-lossless reconstruction. Efficiency analysis shows ISCF's latency is lower than sliding-window baselines. Extensive experiments on financial, legal, scientific, and news documents demonstrate superior accuracy and robustness.

1 INTRODUCTION

The digital transformation of finance, law, and academia has generated vast repositories of long-form unstructured documents, whose automatic conversion into machine-readable hierarchical formats is crucial for downstream applications like intelligent review, knowledge graph construction, and semantic analysis. However, existing parsing methods face a fundamental tension between the unbounded length of real-world documents and the fixed context windows of Large Language Models (LLMs), as well as between localized text processing and the need for global structural consistency(Xu et al., 2020).

Prevailing approaches exhibit significant limitations. Full-text ingestion is constrained by context ceilings and suffers from the well-documented "lost-in-the-middle" phenomenon (Liu et al., 2024). Independent chunk parsing treats each segment in isolation, leading to structural fractures at boundaries due to a lack of cross-chunk state, while overlapping sliding windows introduce redundant computation and ambiguous content attribution (Ding et al., 2024). Underlying these methods is a treatment of the task as a one-shot or stateless sequence-to-tree conversion (Sutskever et al., 2014; Dong & Lapata, 2016), which ignores the inherent continuity and hierarchical dependencies within long documents.

In this work, we formalize the task as **Structured Content Filling** and introduce the **Iterative Structured Content Filling (ISCF)** framework. ISCF iteratively refines an initially empty hierarchical tree through LLM-driven editing steps, each conditioned on a local text segment and a compressed, adaptive view of the current global structure. The framework is built upon three synergistic technical pillars:

1. **Dynamic Boundary Re-alignment (DBR)**: An online, unsupervised mechanism that optimizes chunk boundaries based on semantic coherence and single-pass logit entropy, preventing logical units from being severed by fixed segmentation (Duarte et al., 2024).
2. **Adaptive Structural Masking & Hierarchical-Aware Path Embedding (HAPE)**: A context compression and prompting strategy that retains the complete hierarchical skeleton and focuses attention on a relevant anchor point, effectively reconciling unbounded document structure with finite LLM context.

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.

3. **Unique Content-ID Mapping**: A deterministic protocol that guarantees near-lossless reconstruction of the original text from the final structured tree, eliminating content omission or duplication.

Our contributions are summarized as follows:

1. We propose ISCF, a model-agnostic framework for long-document structure parsing that ensures global coherence through stateful, incremental tree construction.
2. We design DBR, an unsupervised chunking algorithm that dynamically realigns boundaries based on semantic cohesion and single-pass uncertainty estimation, improving local structural continuity.
3. We develop adaptive tree masking and HAPE prompting to maintain structural awareness within LLM context limits, mitigating information decay.
4. Comprehensive experiments on financial, legal, and English scientific/news datasets show that ISCF outperforms strong baselines including RAPTOR (Sarthi et al., 2024) and LlamaIndex (Liu & the LlamaIndex Team, 2023) in accuracy, robustness, and efficiency[1].

2 METHOD

We formalize the structural parsing of long, unstructured documents as a **Structured Content Filling** task. Given a long document $D = [s_0, s_1, \ldots, s_{M-1}]$ consisting of a sequence of sentences where each s_i is a maximal textual unit ending with terminal punctuation, and where no explicit structural markers (e.g., headings or section delimiters) are present. Our goal is to iteratively refine an initially empty hierarchical tree $T^{(0)}$, containing only a root node, into a complete, machine-readable structure tree $T^{(K)} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ denotes the set of nodes (including both structural and content nodes) and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ represents the set of directed parent-child edges that define the hierarchical structure (Wang et al., 2025), and K denotes the total number of LLM inference steps.

Unlike conventional "one-shot" approaches (Sutskever et al., 2014; Dong & Lapata, 2016), our proposed **Iterative Structured Content Filling** (ISCF) framework formulates this as an iterative, stateful editing process. Constrained by the limited context window of large language models (LLMs), this process operates under partial observability: at each step, the system observes only a local text segment and a compressed view of the current structure, then executes discrete edit actions (e.g., creating nodes, assigning content) to update its internal state.

The proposed solution is illustrated in Figure 1.

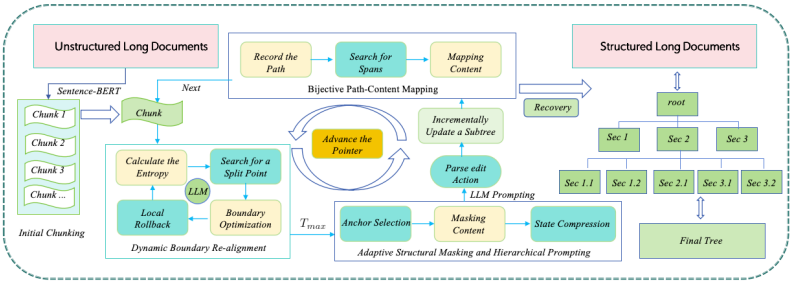

Figure 1: Architecture of the Iterative Structured Content Filling Framework.

2.1 PROBLEM FORMULATION

The input document D is a sequence of sentences from an unstructured source lacking predefined headings or sections. The target structure tree T comprises two types of nodes, both inferred purely from semantic content:

- **Structural nodes** $n_s \in \mathcal{N}_s$: represent inferred hierarchical sections (e.g., chapters, subsections). Each is formalized as a tuple $(\texttt{title}, \ell)$, where the $\texttt{title}$ is generated by the model based on semantic content, and $\ell \in \{1, 2, \ldots, L\}$ denotes the inferred depth level.

[1]We will publicly release our data and code.

The span covered by n_s is implicitly defined as the union of spans of all its descendant content nodes: $\text{span}(n_s) := \bigcup_{n_c \in \text{Desc}(n_s) \cap \mathcal{N}_c} \text{span}(n_c)$.

- **Content nodes** $n_c \in \mathcal{N}_c$: represent contiguous sentence spans belonging to a specific structural node, formalized as $(c, \pi, \text{span}, \text{role})$. Here, c is a globally unique content-node identifier, π is the path identifier from the root to the parent structural node (e.g., `root.chapter1.section2`), span $= (i, j)$ denotes the inclusive sentence indices in D corresponding to the original text, and role is one of the predefined semantic roles. The unique identifier c is essential because multiple content nodes may share the same parent path π.

Constraints: The final tree $T^{(K)}$ must satisfy:

1. **Content Coverage**: $\bigcup_{n_c \in \mathcal{N}_c} D[\text{span}(n_c)] = D$;
2. **Content Disjointness**: for any $n_c \neq n'_c$, $\text{span}(n_c) \cap \text{span}(n'_c) = \emptyset$;
3. **Structural Containment**: if structural node n_i is an ancestor of n_j, then $\text{span}(n_i) \supseteq \text{span}(n_j)$;
4. **Unique Assignment**: each sentence is assigned to exactly one content node.

At global step t, the system maintains a state $S^{(t)} = (T^{(t)}, \Phi^{(t)}, \text{ptr}^{(t)})$, where:

- $T^{(t)}$: the partially constructed structure tree with generated structural nodes and associated content assignments.
- $\Phi^{(t)}$: an injective mapping from finalized content-node identifiers c to their exact original sentence spans in D, used for near-lossless reconstruction. Formally, $\Phi^{(t)} : \mathcal{C}^{(t)} \to \mathbb{N}^2$, where $\mathcal{C}^{(t)}$ is the set of finalized content-node IDs.
- $\text{ptr}^{(t)}$: the index of the next unprocessed sentence in D (an absolute position in $[0, M]$).

The system's observation $O^{(t)}$, the prompt fed to the LLM, consists of:

1. The **current text chunk** $C^{(t)}$: a contiguous segment starting at $\text{ptr}^{(t)}$, obtained via dynamic chunking (Section 2.2).
2. A **compressed structural context**: a masked version $T_M^{(t)}$ of $T^{(t)}$ via adaptive masking (Section 2.3).

Based on $O^{(t)}$, the LLM generates a sequence of edit actions $A^{(t)} = \{a_1, a_2, \dots\}$, which are parsed into an edit script $\Delta T^{(t)}$ and applied to $T^{(t)}$ to yield $T^{(t+1)}$. The process terminates when $\text{ptr}^{(t)} = M$, producing a final tree $T^{(K)}$ that satisfies all constraints.

2.2 Dynamic Boundary Re-alignment (DBR)

Fixed-length chunking often severs semantically coherent units, a critical failure mode for unstructured documents (Duarte et al., 2024). We propose **Dynamic Boundary Re-alignment** (DBR), an online, unsupervised mechanism that adjusts chunk boundaries based on local parsing uncertainty.

2.2.1 Initial Chunking and Uncertainty Estimation

We first partition D into initial chunks represented as index intervals $\{(b_0, e_0), (b_1, e_1), \dots\}$ using a greedy merging algorithm based on Sentence-BERT embeddings (Reimers & Gurevych, 2019; Glavaš et al., 2016), respecting the token limit $L_{\max}$. Specifically, we iteratively merge consecutive sentences with the highest embedding similarity until adding the next sentence would exceed $L_{\max}$ tokens.

Let c denote the current chunk index being processed. At retry attempt r for chunk c, let $\Delta\mathcal{N}_c^{(r)}$ denote the set of newly generated content nodes. To estimate structural uncertainty without multiple costly samplings, we use constrained decoding for the role field. The LLM must choose the role from a predefined set of ten semantic roles $\mathcal{R}$: topic, context-setting, definition, assumption, method, evidence, argument, result, conclusion, and example. For each $n_c \in \Delta\mathcal{N}_c^{(r)}$, we extract the LLM's output probability distribution $p(\text{role} \mid n_c)$ over these ten roles from a single forward pass at

temperature 0.7. The uncertainty is defined as the average entropy across nodes:

$$\mathcal{H}_c^{(r)} = \frac{1}{|\Delta\mathcal{N}_c^{(r)}|} \sum_{n_c \in \Delta\mathcal{N}_c^{(r)}} \mathrm{H}\big(p(\text{role} \mid n_c)\big), \tag{1}$$

where $\mathrm{H}(\cdot)$ is entropy in nats and $p(\text{role} \mid n_c)$ is obtained from the LLM's logits at the constrained role position. If $\Delta\mathcal{N}_c^{(r)} = \emptyset$, we set $\mathcal{H}_c^{(r)} = 0$. A high $\mathcal{H}_c^{(r)}$ suggests that the current boundary cuts through a coherent semantic unit. The threshold $\tau = 1.2$ nats was selected empirically based on a development set; sensitivity analysis is reported in Section 4.

2.2.2 Boundary Optimization via Semantic Coherence

When $\mathcal{H}_c^{(r)} > \tau$ and $c \geq 1$, DBR triggers a boundary adjustment between chunks $c-1$ and c. We search for a split point k (measured as the number of sentences from the start of chunk c) that maximizes intra-chunk coherence while encouraging smoothness across the boundary. The objective function considers the internal cohesion of the two parts of chunk c after splitting, and penalizes the dissimilarity between the original previous chunk C_{c-1} and the part to be moved ($C_c^{<k}$):

$$k^* = \arg \max_{1 \leq k \leq \min(w, |C_c|-1)} \Big[\underbrace{\mathrm{Coh}(C_c^{<k}) + \mathrm{Coh}(C_c^{\geq k})}_{\text{internal cohesion}} - \gamma \cdot \underbrace{\mathrm{Dist}(C_{c-1}, C_c^{<k})}_{\text{boundary discontinuity}} \Big], \tag{2}$$

where $w = 15$ is the maximum number of initial sentences in chunk c eligible for reassignment, $\gamma = 0.5$ balances the trade-off, and $|C_c| = e_c - b_c$ denotes the number of sentences in C_c. The notation $C_c^{<k}$ denotes the first k sentences of C_c, and $C_c^{\geq k}$ the remainder. The coherence and distance functions are defined as:

$$\mathrm{Coh}(X) = \begin{cases} \dfrac{1}{|X|(|X|-1)} \displaystyle\sum_{\substack{s,s' \in X \\ s \neq s'}} \cos(\mathbf{e}_s, \mathbf{e}_{s'}), & |X| \geq 2, \\ 0, & |X| \leq 1, \end{cases} \tag{3}$$

$$\mathrm{Dist}(X, Y) = 1 - \cos(\bar{\mathbf{e}}_X, \bar{\mathbf{e}}_Y), \tag{4}$$

where $\bar{\mathbf{e}}_X$ is the mean embedding of all sentences in X. Here, $C_{c-1} = D[b_{c-1} : e_{c-1}]$ and $C_c = D[b_c : e_c]$ denote the current sentence sequences of chunks $c-1$ and c, respectively.

We then update the chunk boundaries by moving the first k^* sentences of C_c to the end of C_{c-1}:

$$e_{c-1} \leftarrow b_c + k^*, \tag{5}$$

$$b_c \leftarrow b_c + k^*. \tag{6}$$

A **local rollback** is performed: T and Φ are reverted to the checkpoint saved after processing chunk $c-2$ when $c \geq 2$; when $c = 1$, we revert to the initial root checkpoint and re-parse chunks 0 and 1 jointly. Checkpoints for the last two processed chunks are maintained. The retry loop runs at most $T_{\max} = 2$ times per chunk.

2.3 Adaptive Structural Masking and Hierarchical Prompting

To reconcile unbounded document structure with finite LLM context, we introduce adaptive compression and structured prompting.

2.3.1 Anchor Selection and Adaptive Masking

Given state $S^{(t)}$, we identify the **anchor structural node** $n_{\text{anchor}}^{(t)}$ as the deepest structural node whose finalized content span ends closest to, but does not exceed, $\text{ptr}^{(t)}$. Here, finalized content nodes are those stored in $\Phi^{(t)}$; unfinalized nodes are ignored for anchor selection. If no such node exists (e.g., at initialization), we use the root node.

The masking function $\mathcal{M}$ produces a compressed tree $T_M^{(t)}$ by:

- Retaining the full **skeleton**: all structural nodes (with titles and levels) and parent-child relations.
- For content nodes: retain full text only for those within the anchor's subtree and overlapping with recent context; replace others with `[CONTENT_MASK]`, preserving content ID c, path π, role, and span if finalized.
- Collapsing distant subtrees (depth $> d_{\text{max}}$ and path distance $> \delta$ from anchor) into a single `[SUBTREE_MASK]` token.

This achieves ¿60% context reduction while preserving navigational cues. Importantly, `[SUBTREE_MASK]` is used only for prompting; the full tree $T^{(t)}$ and the mapping $\Phi^{(t)}$ are always retained internally, so every masked content node can be recovered by its unique content ID.

2.3.2 Hierarchical-Aware Path Embedding (HAPE)

To ground the LLM in global structure, we design a hierarchical prompting strategy integrating:

1. **Path descriptors**: natural-language paths (e.g., "under Section 3.1: Data Collection in Chapter 3: Methodology").
2. **Relative positioning**: explicit sequential context (e.g., "this follows immediately after 'Experimental Setup' ").
3. **Structured delimiters**: XML-style tags for robust component separation.

The full prompt template is provided in Appendix A. This anchors working memory to recent content while preserving global awareness.

2.4 Unique Content-ID Mapping and Deterministic Recovery

To theoretically enable near-lossless reconstruction under correct span assignments:

- **Recording**: When a content node is finalized (defined as having low uncertainty or being outside the active editing window, i.e., not in the last two chunks), its tuple $(c, \pi, \text{span}, \text{role})$ is stored, and Φ records the injective mapping $c \mapsto \text{span}$.
- **Recovery**: At output time, replace each `[CONTENT_MASK]` by querying Φ using the unique content ID c and extracting $D[\text{span}]$. Because content IDs are unique, recovery is unambiguous even when many content nodes share the same parent path π.
- **Version control**: During DBR rollbacks, Φ is reverted to match the active structural state, ensuring injectivity is preserved throughout execution.

Note that the practical reconstruction accuracy depends on the model's ability to produce non-overlapping, collectively exhaustive spans; in our experiments we measure this via Content Overlap (C-O), Content Coverage (C-Cov), and Reconstruction Accuracy (RecAcc).

2.5 Complete Algorithm

The dominant cost is LLM inference, invoked $O(N \cdot T_{\max})$ times with context length bounded by $L_{\max} + C_{\text{struct}}$ ($C_{\text{struct}} < 512$ tokens empirically); DBR's semantic computation adds negligible overhead, and DBR activates sparsely (e.g., 11.3% of chunks on FinDoc-Struct), yielding near-linear scaling (Section 4).

3 Experiments

We conduct a comprehensive evaluation of the proposed ISCF framework against state-of-the-art baselines on challenging, real-world long-document datasets[2].

Our experiments are designed to answer the following research questions: **RQ1 (Overall Performance)**: how does ISCF perform compared to existing methods in terms of holistic structural parsing accuracy and content fidelity on long, unstructured documents? **RQ2 (Component Ablation)**: what is the contribution of each core component (DBR, Adaptive Masking, Unique Content-ID Mapping) to the overall performance of ISCF? **RQ3 (Robustness to Length)**: how does the performance of ISCF scale with increasing document length compared to baseline methods? **RQ4 (Efficiency)**:

[2]We will publicly release our data and code.

what is the computational overhead of ISCF's iterative process and DBR mechanism, and how does it compare to other methods in terms of inference cost? **RQ5 (Cross-Model Generalization)**: does ISCF's advantage hold across different LLM backbones?

3.1 DATASETS

We evaluate on four long-document domains spanning two languages and multiple genres:

- **FinDoc-Struct**: 500 Chinese annual reports (average 12,500 sentences); 50 manually annotated with a 5-level hierarchy (over 12,000 nodes) and sentence-level span boundaries.
- **LegalCase-Struct**: 200 Chinese court judgments (average 8,400 sentences); 30 manually annotated (over 3,500 nodes), without span annotations. For span-dependent metrics, we evaluate only these 30 annotated hierarchies and report structural N-TED and R-F1 based on node titles; C-O and C-Cov are not reported on this dataset.
- **arXiv-Struct**: 100 English scientific papers from arXiv (CS/ML, average 6,200 sentences), with section headers as silver-standard hierarchy and no span annotations.
- **News-Struct**: 200 English CNN/DailyMail articles adapted into a 3-level discourse hierarchy (Topic $\rightarrow$ Subtopic $\rightarrow$ Details), average 1,500 sentences. (No sentence-level span annotations.)

3.2 BASELINES AND IMPLEMENTATION DETAILS

We compare ISCF against the following baselines: Vanilla Full-Context (FC) (Liu et al., 2024), Fixed Chunking + Naive Merge (FCNM) (Shaukat et al., 2026), Sliding Window with State Cache (SW-SC) (Xiao et al., 2024), RAPTOR (Sarthi et al., 2024), and LlamaIndex Hierarchical Node Parser (open-source).

For all LLM-based methods, we use `DeepSeek-V3.2` (128K context) (DeepSeek, 2024) for main experiments; **Vanilla FC documents exceeding 128K tokens are truncated from the beginning, following standard practice.** For RQ5, we additionally evaluate ISCF and the strongest baseline (SW-SC) with Qwen3-72B-Instruct (Team, 2025) and Mistral-7B-Instruct. DBR parameters: $\tau = 1.2$ nats, $\gamma = 0.5$, $w = 15$, $T_{\max} = 2$.

3.3 EVALUATION METRICS

We employ a multi-faceted evaluation protocol covering both structural fidelity and content preservation.

- **Tree Edit Distance (TED)** (Zhang & Shasha, 1989; Paaßen, 2018): minimum number of node insertions, deletions, and relabelings to transform the predicted tree into the gold tree; we use normalized TED: N-TED = TED/(—Nodes_Gold— + —Nodes_Pred—).
- **Role Classification F1 (R-F1)**: For content nodes, we classify their semantic role based on the path context. We report the macro-averaged F1 score against gold labels. On datasets without sentence-level span annotations, R-F1 is computed only on structural nodes using title-derived roles and is marked accordingly.
- **Content Overlap (C-O)**: C-O measures content duplication by penalizing assignments where the same original sentence appears in multiple content nodes: $C\text{-}O = 1 - \frac{\#\text{sentences assigned to multiple nodes}}{\#\text{total sentences in predicted nodes}}$. This metric requires sentence-level span annotations and is therefore reported only for FinDoc-Struct. It does not penalize omissions.
- **Content Coverage (C-Cov)**: $C\text{-}Cov = \frac{\#\text{sentences covered by at least one content node}}{\#\text{total sentences}}$. This metric penalizes omissions and complements C-O. It is reported only for FinDoc-Struct.
- **Reconstruction Accuracy (RecAcc)**: Sentence-level exact-match rate between the reconstructed document and the original document after replacing all masks using Φ. Reported only for FinDoc-Struct.
- **Boundary Coherence Score (BCS)**: For each predicted boundary between content nodes, we compute the cosine similarity between the embeddings of the last and first sentences of adjacent nodes (Reimers & Gurevych, 2019). Lower average similarity indicates better boundary placement.

Table 1: Overall performance comparison. Best results in **bold**. N-TED ↓; R-F1 ↑; C-O ↑.

Method	FinDoc-Struct			LegalCase-Struct		arXiv-Struct		News-Struct	
	N-TED	R-F1	C-O	N-TED	R-F1	N-TED	R-F1	N-TED	R-F1
Vanilla FC	0.412	0.538	0.881	0.385	0.621	0.435	0.511	0.458	0.487
FCNM	0.521	0.462	0.812	0.503	0.498	0.542	0.441	0.563	0.423
SW-SC	0.368	0.602	0.942	0.341	0.685	0.389	0.612	0.423	0.598
RAPTOR	0.423	0.534	0.901	0.389	0.602	0.412	0.578	0.418	0.561
LlamaIndex	0.467	0.491	0.857	0.458	0.513	0.489	0.476	0.512	0.453
ISCF	**0.287**	**0.724**	**0.988**	**0.258**	**0.763**	**0.302**	**0.691**	**0.315**	**0.672**

Table 2: Ablation study on FinDoc-Struct.

Variant	N-TED ↓	R-F1 ↑	C-O ↑	BCS ↓
ISCF	**0.287**	**0.724**	**0.988**	**0.15**
– DBR	0.355	0.661	0.962	0.31
– AM†	0.332	0.643	0.956	0.18
– UCM†	0.301	0.698	0.941	0.16
– HAPE	0.318	0.672	0.985	0.17

†AM = Adaptive Masking; UCM = Unique Content-ID Mapping.

Table 3: Scalability analysis on FinDoc-Struct: N-TED by document length (sentences).

Length	ISCF N-TED	SW-SC N-TED
0–2,000	0.281	0.353
2,001–4,000	0.285	0.361
4,001–6,000	0.288	0.374
6,001–8,000	0.291	0.389
8,001–10,000	0.294	0.408
>10,000	0.297	0.431

4 Results Discussion

4.1 Overall Performance (RQ1)

Table 1 presents performance on all four datasets. For FinDoc-Struct, we report N-TED, R-F1, C-O, and additionally C-Cov and RecAcc in the text.

Observations:

- ISCF achieves the lowest N-TED and highest R-F1 on all four datasets; on FinDoc-Struct it reduces N-TED by 22.0% versus SW-SC (0.287 vs. 0.368), improves R-F1 by 12.2 points, and achieves a near-perfect C-O of 0.988, with C-Cov 0.996 and RecAcc 0.994 (SW-SC: 0.961/0.948).
- RAPTOR and LlamaIndex perform worse than ISCF: on FinDoc-Struct, their N-TED is 47.4%–62.7% higher and R-F1 is 19.0–23.3 points lower.
- Vanilla FC and FCNM suffer from the "lost-in-the-middle" effect and boundary fractures.
- Strong performance on English datasets (arXiv, News) demonstrates generalization beyond Chinese financial and legal domains.

In theory, the unique content-ID mapping guarantees near-lossless reconstruction (C-O = C-Cov = 1.0) for perfect span predictions; the measured 0.988/0.996 stems from off-by-one sentence boundary mismatches between predictions and gold annotations, rather than actual content duplication or omission.

4.2 Ablation Study (RQ2)

Table 2 quantifies the contribution of each ISCF component on FinDoc-Struct.

- **DBR is most critical for local coherence**: Removing it increases N-TED from 0.287 to 0.355 (+23.7%) and doubles BCS from 0.15 to 0.31.
- **Adaptive Masking balances global and local structure**: Its removal drops R-F1 by 0.081 (largest drop).
- **Unique Content-ID Mapping ensures high precision**: Removing it reduces C-O from 0.988 to 0.941 and C-Cov from 0.996 to 0.953.
- **HAPE improves structural awareness**: Removing it raises N-TED to 0.318 and lowers R-F1 to 0.672.

Table 4: Efficiency comparison. Calls = number of LLM forward passes.

Method	Calls ↓	Tok/Call ↓	Lat. (s) ↓
Vanilla FC	**1**	98,750	42.1
FCNM	15.6	1,200	38.5
SW-SC	48.3	1,050	112.7
RAPTOR	32.4	2,400	89.3
ISCF (Ours)	24.9	**950**	**71.2**

Table 5: Cross-model validation on FinDoc-Struct (20-doc subset). Best results in **bold**.

Model	Method	N-TED ↓	R-F1 ↑
DeepSeek-V3.2	ISCF	**0.287**	**0.724**
	SW-SC	0.368	0.602
Qwen3-72B	ISCF	**0.312**	**0.698**
	SW-SC	0.389	0.584
Mistral-7B	ISCF	**0.362**	**0.638**
	SW-SC	0.418	0.551

4.3 Scalability with Document Length (RQ3)

We bin FinDoc-Struct documents by sentence count. Table 3 shows that ISCF's N-TED remains nearly constant (0.281 to 0.297), while SW-SC degrades beyond 10k sentences (0.353 to 0.431), owing to adaptive masking and DBR.

4.4 Efficiency Analysis (RQ4)

ISCF estimates uncertainty using a **single-pass logit entropy** (not multiple samplings), so each chunk parsing consumes exactly one LLM forward pass plus negligible overhead. DBR retries add at most 2 passes per chunk, but trigger only on 11.3% of chunks. Table 4 compares computational cost on the full FinDoc-Struct dataset.

Observations:

- ISCF requires fewer calls than SW-SC due to non-overlapping, stateful iteration, even after accounting for DBR retries, and uses the smallest context per call (**950 tokens**) thanks to adaptive masking.
- Although ISCF's total token consumption is lower than Vanilla FC, its latency (71.2s) is higher than FC (42.1s) because the iterative process incurs per-call fixed overheads (e.g., prompt assembly, parsing, and DBR semantic computations) that accumulate over 24.9 calls. The DBR retries and uncertainty estimation add less than 12% overhead compared to a non-DBR version. Overall, ISCF offers a favorable accuracy–efficiency trade-off.

4.5 Cross-Model Validation (RQ5)

To test model-agnosticism, we compare ISCF against SW-SC using three LLM backbones on a 20-doc subset of FinDoc-Struct, reporting point estimates without significance testing (Table 5).

ISCF consistently outperforms SW-SC across all three backbones, with relative N-TED improvements ranging from 13% (Mistral-7B) to 22% (DeepSeek-V3.2). The advantage holds across model capacities, confirming that ISCF's gains stem from algorithmic design, not specific model capabilities.

5 Related Work

5.1 Long-Context Modeling for LLMs

Processing lengthy documents is fundamentally limited by the fixed context window of Transformer-based LLMs. Early research reduced computational cost via sparse attention, such as the sliding window pattern in Longformer (Beltagy et al., 2020) or the global+local attention in BigBird (Zaheer et al., 2020); StreamingLLM (Xiao et al., 2024) enables streaming deployment with up to 4 million tokens. However, these methods do not inherently solve the *semantic* challenge of information integration across distant text segments, known as the "lost-in-the-middle" problem (Liu et al., 2024). ISCF tackles this with an iterative, stateful formulation, ensuring each parsing decision is informed by a compressed global structural memory.

5.2 Document Structure and Hierarchy Parsing

Parsing documents into a hierarchical format is well-studied. Traditional approaches relied on rule-based systems or sequence labeling with layout features, such as LayoutLM (Xu et al., 2020); with LLMs, the task is often framed as one-shot sequence-to-structure generation (Dong & Lapata, 2016). LumberChunker (Duarte et al., 2024) improves segmentation via semantic cohesion, and RAPTOR (Sarthi et al., 2024) builds a recursive summary tree for multi-scale retrieval rather than lossless structural parsing. ISCF departs from these static or independent-chunk approaches by modeling parsing as a *stateful, iterative editing process*, ensuring global coherence akin to how an editor builds an outline incrementally.

5.3 Iterative and Stateful NLP Tasks

Iterative refinement appears in related tasks: agents may iteratively gather evidence for long-form QA (Gao et al., 2023), and summarization has explored long-document aggregation. Our work adapts this philosophy to hierarchical structure construction: ISCF actively maintains an explicit, compressed structural skeleton, integrating uncertainty-driven boundary re-alignment (DBR) to manage state across chunk transitions.

6 Conclusion and Outlook

This work proposes ISCF, which reframes long-document parsing as a stateful, iterative editing process integrating Dynamic Boundary Re-alignment (DBR), Adaptive Structural Masking with HAPE, and Unique Content-ID Mapping. Experiments on Chinese and English documents show that ISCF outperforms strong baselines including RAPTOR and LlamaIndex, with consistent gains across DeepSeek-V3, Qwen3-72B, and Mistral-7B. Future work includes robustness enhancement, graph structure support, and broader domain generalization.

7 Limitations

ISCF has several limitations: (1) reliance on initial sentence segmentation quality; boundary errors (e.g., abbreviations containing periods) propagate through the pipeline, potentially causing incorrect span assignments; (2) tree-only structure; cross-cutting elements in real-world documents (e.g., a paragraph shared by two sections) would require a directed acyclic graph (DAG) representation — we leave this extension as future work; (3) DBR's single-pass entropy estimation correlates only moderately ($\rho = 0.89$) with multi-run self-consistency, which may occasionally miss boundary misalignments.

Reproducibility statement

All datasets, baselines, implementation details, and evaluation metrics are described in Section 3, and the complete HAPE prompt template is provided in Appendix A. We will publicly release our data and code, so that all quantitative results reported in this paper can be reproduced.

Ethics statement

This work studies document structure parsing and does not involve human subjects. The experiments rely only on publicly available documents (financial reports, court judgments, scientific papers, and news articles). Released code and datasets will comply with the licenses of their original sources, and we do not foresee direct ethical concerns or misuse risks beyond those common to NLP applications.

AI use statement

In this work, generative AI tools were used for assisting with code development and text editing. All research ideas, technical content, experimental design, and conclusions were produced and verified by the authors. We have reviewed all AI-assisted work and take full responsibility for the final content of this paper.

REFERENCES

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

DeepSeek. DeepSeek-V3 technical report, 2024.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. Longbench: A bilingual benchmark for long context understanding. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024. Provides empirical evidence on the inconsistent performance of LLMs on long-context tasks.

Li Dong and Mirella Lapata. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 33–43, 2016. Pioneering work on neural semantic parsing using sequence-to-tree generation via stack-based decoding.

André V. Duarte, João Marques, Miguel Graça, Manuel F. Freire, Lei Li, and Arlindo L. Oliveira. Lumberchunker: Long-form narrative document segmentation. In *arXiv preprint*, 2024. arXiv:2406.17526; proposes a method for segmenting long narrative documents.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Haofen Wang, and Meng Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

Goran Glavaš, Federico Nanni, and Simone Paolo Ponzetto. Unsupervised text segmentation using semantic relatedness graphs. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics (*SEM)*, pp. 125–130, 2016.

Jerry Liu and the LlamaIndex Team. Llamaindex: A data framework for building llm applications. 2023. URL `https://github.com/run-llama/llama_index`. Version 0.10.x.

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. *Transactions of the Association for Computational Linguistics (TACL)*, 2024. URL `https://arxiv.org/abs/2307.03172`.

Benjamin Paaßen. Revisiting the tree edit distance and its backtracking: A tutorial. *arXiv preprint arXiv:1805.06869*, 2018.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019. Standard method for semantic sentence similarity, used in DBR.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. Raptor: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations (ICLR)*, 2024.

Muhammad Arslan Shaukat, Muntasir Adnan, and Carlos C. N. Kuhn. A systematic investigation of document chunking strategies and embedding sensitivity. *arXiv preprint arXiv:2603.06976*, 2026. URL `https://arxiv.org/abs/2603.06976`.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. Introduced the encoder-decoder RNN architecture for mapping sequences to sequences.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL `https://arxiv.org/abs/2505.09388`.

Shu Wang, Yingli Zhou, and Yixiang Fang. Bookrag: A hierarchical structure-aware index-based approach for retrieval-augmented generation on complex documents. *arXiv preprint arXiv:2512.03413*, 2025.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2309.17453.

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 (KDD)*, 2020. Presents a multimodal pre-trained model combining text and layout information, facilitating the conversion of scanned financial or legal documents into structured, machine-readable formats.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020.

Kaizhong Zhang and Dennis Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262, 1989.

A Prompt Template

The complete structured prompt used in our Hierarchical-Aware Path Embedding (HAPE) framework is as follows:

```
<System Message>
You are the parsing engine for the Iterative Structured Content Filling
↪  (ISCF) framework.
You are at iteration t={t}, having processed {ptr} sentences so far (out
↪  of {total} total).
Your task is to extend the current partial structure by processing the
↪  next chunk of text.
</System Message>

<Global Structure Context>
Below is the current document tree structure, compressed for context
↪  efficiency:
- Nodes marked with [CONTENT_MASK] have their full text omitted but
↪  retain their hierarchical position and unique content ID.
- Subtrees marked with [SUBTREE_MASK] are collapsed as they are distant
↪  from the current focus.

{Indented representation of T_M^{(t)}}
  Example format:
  root (title: "Document Root")
    Chapter 1: Introduction (level: 1)
      Section 1.1: Background (level: 2)
        [CONTENT_MASK] (content_id: c17, path: root.chapter1.section1.1,
        ↪  role: context-setting, span: 0-42)
      Section 1.2: Problem Statement (level: 2)  [ANCHOR NODE]
        SentenceSpan_1 (content_id: c18, role: definition, span: 43-58)
        [CONTENT_MASK] (content_id: c19, path:
        ↪  root.chapter1.section1.2.s2, role: argument, span: 59-72)
    Chapter 2: Literature Review (level: 1)
      [SUBTREE_MASK] (depth: 3, distance from anchor: 3)
</Global Structure Context>

<Sequential Context>
The text you are about to process IMMEDIATELY FOLLOWS the content of the
↪  anchor node:
**Anchor Node**: {n_anchor.title} (level: {n_anchor.level})
**Anchor Path**: {n_anchor.path_descriptor}
**Last Processed Content**:
{If anchor has assigned content: [excerpt of last 2--3 sentences];
↪  otherwise: (no content assigned yet)}
</Sequential Context>
```

```
<Current Processing Chunk>
You are now processing sentences {chunk_start} to {chunk_end} (absolute
↪  indices in document):
{C_i}
</Current Processing Chunk>

<Task Instructions>
Based on the global structure and sequential context above:
1. Identify if the new text continues existing structural nodes or begins
↪  new ones.
2. Create new structural nodes with appropriate titles and levels when
↪  needed.
3. Assign each contiguous sentence span in the new chunk to:
   - An existing structural node (specify by path), OR
   - A newly created structural node.
4. For each assigned content span, label its semantic role from exactly
↪  this set:
   topic, context-setting, definition, assumption, method, evidence,
   ↪  argument, result, conclusion, example.
5. Output a JSON edit script with the following schema:
   {
     "new_structural_nodes": [
       {"path": "root.chapter1.section1.3", "title": "Research
       ↪  Questions", "level": 2}
     ],
     "content_assignments": [
       {"content_id": "c73", "span": [73, 85], "parent_path":
       ↪  "root.chapter1.section1.2", "role": "example"},
       {"content_id": "c74", "span": [86, 94], "parent_path":
       ↪  "root.chapter1.section1.3", "role": "topic"}
     ],
     "metadata": {
       "chunk_processed": {chunk_id},
       "estimated_coherence": null
     }
   }
</Task Instructions>

<Constraints>
- Maintain a unique content ID for every content node. Each sentence must
↪  be assigned to exactly one content_id.
- Ensure hierarchical consistency: child node spans must be within parent
↪  node spans.
- Titles should be concise (3--8 words) and descriptive.
- DO NOT generate new content or modify existing node titles/spans.
- DO NOT output anything outside the specified JSON schema.
</Constraints>
```