PharmaSUG 2026 - Paper TT-394

Structure-Preserving Preprocessing of Clinical Documents for Large Language Model Analysis

Zun Wang, R & G US Inc;
Juntao Yan, Eli Lilly and Company

ABSTRACT

Clinical trial documents are essential to drug development but are often lengthy and labor intensive to align or compare within or across studies due to strict traceability and governance requirements. Although large language models (LLMs) are increasingly used to improve document analysis efficiency, directly applying LLMs to flat text presents practical challenges. Document length can exceed model context limits, and loss of section boundaries prevents LLMs from recognizing the inherent hierarchical structure of clinical documents, compromising both accuracy and traceability.

To address these challenges, this paper introduces a structure-preserving preprocessing workflow that reconstructs clinical documents (e.g., DOCX and PDF files) into an explicit hierarchical representation prior to LLM analysis. By chunking documents according to semantic sections rather than processing full documents, the workflow supports more efficient LLM interaction while preserving section-level traceability. This design also reduces the risk of mixing information across unrelated sections, keeping outputs grounded in clearly scoped context.

In practice, this approach enables faster and more reliable clinical document review through targeted, section-level analysis. Limiting analysis to relevant sections reduces the number of LLM calls required, improving processing efficiency and turnaround time. The workflow applies across common clinical trial documents, including protocols and SAPs, and supports practical use cases such as version control and protocol–SAP alignment within a study. Explicit section alignment allows differences to be evaluated within matched structural context, reducing manual review effort during iterative document updates.

INTRODUCTION

Clinical trial documents, including protocols, statistical analysis plans (SAPs), and amendments, are revised iteratively throughout a study and reviewed through well-defined document sections. In routine clinical workflows, multiple document types and versions must remain aligned while supporting traceability, version control, and section-level interpretation in regulated environments (U.S. Food and Drug Administration 2024). In routine practice, clinical document review often involves reconciling protocol objectives with SAP methods and downstream outputs such as tables, listings, and figures (TLFs) across multiple document versions through section-level comparison rather than whole-document review. Ensuring consistency across documents and across versions therefore needs great manual effort. Thus, clinical study teams rely heavily on manual review to locate relevant content, interpret contextual changes, and assess potential downstream impact, often under tight study timelines.

Although LLMs have shown promise for analyzing narrative text, their direct application to clinical trial documents presents practical challenges in regulated settings. Long clinical documents frequently approach practical model context limits, making whole-document prompting less reliable in practice (Gómez Cabello et al. 2025). In clinical settings, this limitation has been associated with the generation of plausible yet unsupported responses, commonly referred to as hallucinations, which can lead to deviations from established practices when task context is incomplete or fragmented. More importantly, treating an entire document as a single input obscures the hierarchical structure that encodes much of its meaning (Yang et al. 2016). When hierarchical structure is not preserved, LLM-generated outputs become difficult to attribute to specific source sections, making it difficult to find origins of their supporting text, particularly in the analysis of long documents (Anh Hoang et al. 2025).

These limitations are particularly critical in clinical review workflows, where the significance of a change depends not only on its content but also on its location within the document hierarchy. Amendments and revisions are often localized to specific sections, and reviewers typically seek to understand their impact

without reanalyzing the entire document. Study teams often have to review large documents multiple times to confirm that updates to key elements, such as endpoints or imputation rules, are consistently reflected across documents. As a result, whole-document or unstructured text-based approaches do not readily support localized, version-aware review or re-execution in controlled clinical workflows.

This paper addresses the structural mismatch between how clinical documents are reviewed in real life and how LLM-based analysis is typically applied. By organizing inputs around document structure prior to LLM interaction, the workflow supports traceable, section-aligned analysis in regulated clinical environments. The following section describes the methodology that operationalizes this perspective.

METHODOLOGY OVERVIEW

WORKFLOW OVERVIEW

The proposed methodology reconstructs the hierarchical section structure of large clinical documents and uses this structure as the primary unit for downstream analysis. The workflow first detects document headings and rebuilds the section hierarchy to create a structural map of the document. Each reconstructed section is assigned a section key and linked to its parent sections, preserving the full hierarchical context. Text content is then merged into section-aligned chunks that retain their associated section identifiers and upper-level section names. These context-preserving chunks are subsequently provided to the LLM for theme identification and reasoning while maintaining awareness of their hierarchical location within the document. The resulting annotations remain aligned with the reconstructed section structure, enabling traceable and context-aware downstream analyses. Figure 1 summarizes the overall workflow. The following section outlines the design principles that guide this structure-preserving analysis framework.

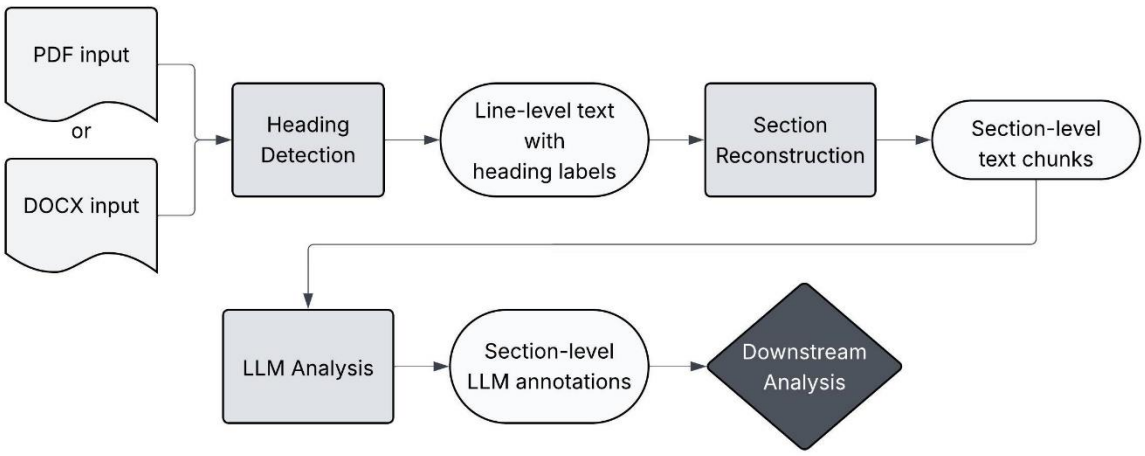

Figure 1. Overview of the Section-Based Document Reconstruction and LLM Analysis Workflow

DESIGN PRINCIPLES AND SCOPE

A key design principle of the proposed methodology is the separation between document structure recovery and LLM-based interpretation. Structural preprocessing is performed prior to any model interaction and operates independently of LLM behavior. During this stage, document headings are detected, and the hierarchical section structure is reconstructed, establishing stable section boundaries and contextual relationships across the document.

Because document structure is established before model interaction, LLM analysis operates on clearly scoped section-level text segments rather than on unstructured document text. This design reduces

ambiguity arising from unclear boundaries and preserves the contextual relationships defined by the document hierarchy.

The methodology is intentionally model-agnostic. It does not rely on specialized model capabilities beyond basic text input and output, allowing different LLMs to be substituted without modifying the structural preprocessing pipeline. By treating document structure as a first-class element of the processing pipeline, the approach provides a general framework for applying LLM-based analysis to hierarchical clinical documents.

ALGORITHM AND IMPLEMENTATION DETAILS

The methodology is conceptually format-independent. However, this study presents one practical implementation supporting DOCX and PDF inputs. For DOCX documents, heading detection leverages native Word heading styles, which provide explicit structural information for reconstructing section hierarchies. For PDF documents, where explicit structural markup is often absent, heading levels are inferred using heuristic analysis of structural cues such as bookmarks, visual layout characteristics, and text similarity. Although heuristic processing introduces additional computational cost, it enables robust structure recovery across heterogeneous PDF documents commonly encountered in clinical workflows.

DOCX HEADING DETECTION

For DOCX inputs, section structures are recovered using native Word heading styles. The document is processed sequentially, paragraph by paragraph. For each paragraph, its text content and associated paragraph style are extracted. Paragraphs assigned to Word heading styles are treated as section headings and mapped directly to their corresponding heading levels.

Paragraphs identified as headings define section boundaries, while non-heading paragraphs are retained as body text. As the document is traversed, each detected heading updates the current section context, which is propagated to subsequent paragraphs until a new heading is encountered. Through this process, a line-level representation is constructed in which each text line is annotated with its heading status and associated section path.

Because DOCX files encode document structure explicitly through heading styles, this approach provides a stable and reproducible mechanism for section hierarchy reconstruction.

```
word_heading_detection <- function(doc_path, max_levels = 10, trailing_dot =
FALSE) {

  # 1) Read DOCX XML
  tmp_dir   <- unzip_docx(doc_path)
  doc_xml   <- read_xml(file.path(tmp_dir, "word/document.xml"))
  p_nodes   <- xml_find_all(doc_xml, ".//*[local-name()='p']")

  # 2) Extract paragraph text + style
  text   <- map_chr(p_nodes, paragraph_visible_text) # visible text only
  pstyle <- map_chr(p_nodes, paragraph_style_id) # read the paragraph's
style identifier

  # 3) Infer heading level from style name (e.g., "Heading 1" -> 1)
  heading_level <- map_int(pstyle, style_to_level) # Use regex to find
Heading N pattern
  is_heading    <- !is.na(heading_level)

  # 4) Keep only visible (non-empty) paragraphs
  df <- tibble(text = text, heading_level = heading_level, is_heading =
is_heading) |>
    filter(text != "")
```

```
  # 5) Add hierarchical numbering to headings (e.g., 1.2.3 Title)
  df$text_with_numbering <- add_heading_numbers(
    text = df$text,
    heading_level = df$heading_level,
    max_levels = max_levels,
    trailing_dot = trailing_dot
  )

  # 6) Return line-level table for downstream structure recovery
  df |> select(text, text_with_numbering, is_heading, heading_level)
}

# Call function

Text_df <- word_heading_detection(doc_path)
```

Program 1. Word Document Heading Detection Logic

PDF HEADING INFERENCE

Unlike DOCX files, PDF documents typically lack explicit semantic markers for document structure. For this reason, section structure recovery for PDF inputs is treated as an inference task that integrates multiple weak structural signals. Document outline information provides an initial cue for heading detection, while line-level layout features extracted from the rendered PDF help infer section headings.

First, bookmark (outline) entries are extracted and treated as candidate section titles with associated hierarchical levels. These outline entries provide high-level structural guidance but do not necessarily correspond directly to rendered text. In parallel, the PDF content is parsed into a sequential stream of text lines annotated with layout attributes such as vertical position and line height. Lines likely originating from tables of contents are identified and excluded to reduce spurious matches.

Outline titles are then aligned to nearby text lines using normalized text similarity, augmented by layout-based scoring that favors visually prominent lines. Matching proceeds sequentially through the document to preserve ordering constraints and reduce cross-section misalignment. When a match satisfies predefined similarity criteria, the corresponding line is marked as a heading and assigned the outline-derived heading level. If outline information is unavailable or no suitable match is identified, the document is treated as unstructured text without inferred headings.

This process yields a line-level representation annotated with inferred heading indicators and levels, providing an interpretable approximation of section hierarchy for downstream chunk construction.

```
pdf_heading_inference <- function(pdf_path) {

  # 1) Extract document outline (bookmarks) as candidate headings
  outline <- extract_pdf_outline(pdf_path)   # title + hierarchical level

  # 2) Extract line-level text with layout attributes
  lines <- extract_pdf_lines(pdf_path)        # page, line_idx, text,
height, position

  # 3) Remove table-of-contents pages or non-body regions
  body_lines <- filter_non_toc_lines(lines)

  # 4) Normalize text for robust matching
  body_lines$text_norm <- normalize_text(body_lines$text)
  outline$title_norm   <- normalize_text(outline$title)
```

```
  # 5) Sequentially align outline titles to nearby lines
  heading_hits <- match_outline_to_lines(
    outline      = outline,
    lines        = body_lines,
    similarity   = text_similarity_score,
    layout_score = heading_visual_score,
    order_safe   = TRUE
  )

  # 6) Assign inferred heading levels to matched lines
  body_lines <- body_lines |>
    left_join(heading_hits, by = c("page", "line_idx")) |>
    mutate(is_heading = !is.na(heading_level))

  # 7) Return line-level table with inferred structure
  body_lines |>
    select(text, is_heading, heading_level)
}
# Call function

Text_df <- pdf_heading_inference(pdf_path)
```

Program 2. PDF Heading Inference Logic

SECTION-AWARE CHUNK CONSTRUCTION

Following section structure recovery, the workflow constructs section-aligned text chunks for model processing. Each line of text is assigned a section path derived from the reconstructed heading hierarchy. As the document is processed sequentially, each line inherits the section path defined by the most recent heading, ensuring that all text remains associated with its enclosing sections. Adjacent lines sharing the same section path are grouped and merged into semantically coherent chunks. To preserve traceability when chunks are analyzed independently, the corresponding section headers are embedded directly within the chunk text so that the hierarchical context remains visible during model processing.

Chunk sizes are controlled to balance contextual completeness with model input constraints. Within each section, adjacent text blocks are merged in document order until a predefined size threshold is reached. This prevents excessively large chunks that may dilute section-specific context or exceed model limits. If a merged block becomes too large, it is further divided into smaller fragments. At the same time, overly small fragments are avoided to preserve semantic continuity within the section.

The resulting chunks remain aligned with their originating section paths and are passed to downstream annotation.

```
build_section_chunks <- function(text_df,
                                 max_chars = 1200,
                                 min_chars = 300) {

  # text_df columns:
  #   text, heading_level, section_1 ... section_k

  # 1) Construct a unique section key from the section path
  sec_cols <- grep("^section_", names(text_df), value = TRUE)

  text_df$section_key <- apply(
    text_df[, sec_cols, drop = FALSE],
    1,
    function(x) paste(ifelse(is.na(x), "", x), collapse = " | ")
  )
```

```
  # 2) Embed section headers into line-level text
  text_df$chunk_text <- mapply(
    function(txt, secs) {
      header <- paste(
        paste0("[Section ", seq_along(secs), "] ", secs),
        collapse = "\n"
      )
      header <- stringr::str_squish(header)
      if (header == "") txt else paste(header, txt, sep = "\n\n")
    },
    text_df$text,
    text_df[, sec_cols, drop = FALSE],
    SIMPLIFY = FALSE
  )

  # 3) Merge adjacent lines within the same section
  merge_within_section <- function(df) {
    out <- character()
    cur <- ""

    for (txt in df$chunk_text) {
      if (nchar(cur) == 0) {
        cur <- txt
      } else if (nchar(cur) + nchar(txt) <= max_chars) {
        cur <- paste(cur, txt, sep = "\n\n")
      } else {
        out <- c(out, cur)
        cur <- txt
      }
    }
    if (nchar(cur) > 0) out <- c(out, cur)
    tibble::tibble(chunk_text = out)
  }

  merged <- text_df |>
    dplyr::group_by(section_key) |>
    dplyr::group_modify(~ merge_within_section(.x)) |>
    dplyr::ungroup()

  # 4) Split oversized chunks to enforce hard size limits
  split_long <- function(x) {
    if (nchar(x) <= max_chars) return(x)
    starts <- seq(1, nchar(x), by = max_chars)
    ends   <- pmin(starts + max_chars - 1, nchar(x))
    substr(x, starts, ends)
  }

  merged |>
    dplyr::mutate(chunk_text = purrr::map(chunk_text, split_long)) |>
    tidyr::unnest(chunk_text)

}
```

Program 3. Chunk Reconstruction Algorithm

LLM INTERACTION AND ANNOTATION

After section-aware chunk construction, the workflow applies model-based annotation to each chunk independently. Prior to annotation, users define several analysis themes that determine the scope of the task (e.g., endpoints, estimands, or imputation methods). These themes guide the model toward targeted extraction and reasoning rather than unconstrained summarization, allowing the analysis to focus on specific aspects of interest within the document.

Each chunk is embedded into a standardized prompt that incorporates the selected theme and constrains both the analysis objective and output format. Because the chunk text retains its section headers and hierarchical context, the resulting annotation remains linked to its originating section path.

To support stable large-scale execution, practical implementations may incorporate operational safeguards such as request retry, rate-limit handling, and result caching. These mechanisms improve execution robustness but do not alter the core workflow.

```
annotate_chunks_with_llm <- function(chunks_df) {

  # Each row in chunks_df represents a single section-aligned text chunk
  # produced by the chunk construction step.

  annotate_one <- function(text_chunk) {

    # Build a standardized prompt that:
    # - defines the annotation task
    # - constrains the output to a fixed schema
    # - limits the model to section-scoped interpretation
    prompt <- build_annotation_prompt(text_chunk)

    # Submit the prompt to the LLM and receive a structured response.
    # The response is expected to contain predefined fields (e.g., theme,
reason).
    result <- call_llm(prompt)

    # Return only the structured annotation fields
    list(
      theme  = result$theme,
      reason = result$reason
    )
  }

  # Apply LLM annotation independently to each chunk.
  # Chunk-level isolation preserves traceability and avoids cross-section
inference.
  chunks_df |>
    dplyr::mutate(
      annotation = purrr::map(chunk_text, annotate_one),

      # Extract structured fields from the LLM response
      theme  = purrr::map_chr(annotation, "theme"),
      reason = purrr::map_chr(annotation, "reason")

    ) |>

    dplyr::select(-annotation)

}
```

Program 4. LLM Implementation

APPLICATION AND USE CASES

HIERARCHY-AWARE LLM ANALYSIS OF CLINICAL DOCUMENTS

To illustrate the practical application of the proposed framework, a simulated SAP was processed through the complete analysis pipeline. The document includes hierarchical headings, derivation rules, analysis population definitions, statistical methodologies, and representative tables typical of clinical trial documentation.

After structure reconstruction and section-aligned chunking, each analytical unit retains its hierarchical section context, allowing the model to interpret content such as statistical hypotheses and endpoint analyses within their appropriate document scope. Example outputs are shown in Table 1.

Text	Is_Heading	Heading_ Level	Section_1	Section_2	Section_3
1 Introduction	TRUE	2	Statistical Analysis Plan	1 Introduction	
1.1 Purpose	TRUE	3	Statistical Analysis Plan	1 Introduction	1.1 Purpose
This Statistical Analysis Plan (SAP) describes the planned statistical analyses for a simulated clinical study…	FALSE		Statistical Analysis Plan	1 Introduction	1.1 Purpose

Table 1. Reconstructed Section Hierarchy from a Pseudo SAP

CONTEXT-PRESERVED CHUNKING FOR LLM TASKS

Following hierarchy reconstruction, the SAP content was partitioned into section-aligned chunks. Each chunk includes its full section path (e.g., Section 5 → Section 5.2 → Section 5.2.1), which is embedded directly within the chunk text supplied to the model. This preserves the hierarchical context associated with each analytical unit. Table 2 illustrates an example chunk in which section headers are embedded before the text content.

Preserving section boundaries helps prevent cross-section inference, where identical terminology (e.g., “baseline,” “responder,” or “analysis population”) may carry different meanings across different parts of the document. When processed by the model, the resulting annotations remain explicitly linked to their originating section paths, enabling traceable interpretation of derivation logic and analytical rules. Table 3 presents example outputs demonstrating theme identification and reasoning derived from section-aware chunks.

Section_Key	Chunk_Text
Statistical Analysis Plan | 7 Handling of Missing Data | 7.2 Sensitivity Analyses | 7.2.2 Multiple Imputation	[Section 1] Statistical Analysis Plan [Section 2] 7 Handling of Missing Data [Section 3] 7.2 Sensitivity Analyses [Section 4] 7.2.2 Multiple Imputation The imputation model will include treatment group, geographic region, visit, baseline Endpoint X value, and relevant demographic covariates (including age and sex) as predictors. Continuous variables will be imputed using predictive mean matching...

Table 2. Section-Aware Chunk with Embedded Hierarchical Context Before LLM Submission

Chunk_Text	Theme	Reason
[Section 1] Statistical Analysis Plan [Section 2] 7 Handling of Missing Data [Section 3] 7.2 Sensitivity Analyses [Section 4] 7.2.2 Multiple Imputation The imputation model will include treatment group, geographic region, visit, baseline Endpoint X value, and relevant demographic covariates...	Statistical Method	The excerpt describes the use of multiple imputation models for handling missing data, including predictive mean matching for continuous variables and logistic regression for categorical variables, with separate models by treatment group.
[Section 1] Statistical Analysis Plan [Section 2] 7 Handling of Missing Data [Section 3] 7.2 Sensitivity Analyses [Section 4] 7.2.2 Multiple Imputation A total of 50 imputed datasets will be generated. Each imputed dataset will be analyzed using the same ANCOVA model specified for the primary efficacy analysis...	Statistical Method	A total of 50 imputed datasets will be analyzed using ANCOVA, with results combined using Rubin's rules for overall estimates and p-values.

Table 3. LLM Output Demonstrating Interpretation of Derivation Rules with Explicit Section-Level Traceability

SECTION-ALIGNED DOCUMENT COMPARISON TOOL

Building on the section-traceable annotations illustrated in Table 3, the framework also supports structured comparison of clinical study documents. Because each analytical unit retains its section key, theme, and reasoning summary, these annotations can serve as inputs for document comparison while preserving traceability to the original document structure. To demonstrate this capability, a lightweight web-based application was developed to support section-aligned comparison of protocols and SAPs, as well as version tracking within a single document type.

The application compares annotated content associated with shared themes. Available themes are automatically detected, allowing users to select specific themes of interest and focus the comparison on particular components of study design or statistical analysis.

Two operational modes are supported.

Consistency Check Mode is designed for cross-document alignment scenarios, such as protocol–SAP comparison. Content associated with the same theme is evaluated together to identify potential factual or numerical inconsistencies. By analyzing related content in batches, the system reduces the number of model calls while preserving sufficient context to detect meaningful differences.

Document Version Control Mode supports comparison across different versions of the same document. In this mode, individual reasoning items are compared using semantic matching to identify three types of outcomes: matched content with no substantive change, discrepancies reflecting modified information, and unique content appearing in only one version. Because this mode performs finer-grained comparisons across individual items, it requires additional processing time.

In both modes, section-aligned content is evaluated by a language model to determine whether information is consistent across documents. Results are categorized as match, discrepancy, or unique content, with each finding linked to the associated theme and explanation.

Use Case 1: Protocol–SAP Alignment

A simulated protocol and SAP were compared using Consistency Check Mode, which evaluates theme-aligned content across documents to identify potential factual or numerical inconsistencies.

As illustrated in Display 1, the system identifies several discrepancies across study themes. For example, the number of analysis populations differs between documents, the specified statistical significance level changes from α = 0.05 to α = 0.15, and the planned number of study visits is inconsistent.

Because the comparison operates on section-aligned analytical units, discrepancies are evaluated within their appropriate document scope.

theme	issue_type	description	file1_detail	file2_detail
Analysis population	Discrepancy	Number of analysis populations defined	4 distinct study populations identified for the analysis	Two analysis populations defined (ITT and Safety)
Statistical method	Discrepancy	Alpha level (statistical significance threshold) differs between files	Two-sided alpha level of 0.05 (excerpt 7)	Two-sided alpha = 0.15 level (excerpt 2)
Study design	Discrepancy	Total number of subjects to be randomized differs between files	Approximately 120 subjects randomized 1:1	Approximate total enrollment of 140 subjects randomized 1:1
Study design	Discrepancy	Number of study visits differs - File 2 includes an additional follow-up visit not present in File 1	5 scheduled visits: Screening, Baseline, Week 4, Week 8, Week 12	6 visits: Screening, Baseline/Randomization, Week 4, Week 8, Week 12, and Follow-up (Week 13 ±3 days)

Display 1. Example Output from Consistency Check Mode Highlighting Discrepancies Between a Protocol and an SAP Across Multiple Study Themes

Use Case 2: Document Version Tracking

Two sequential SAP versions were compared using Document Version Control Mode, which performs semantic matching across reasoning items to identify unchanged content and meaningful revisions.

As shown in Display 2, the system detects a discrepancy in the planned sample size: one version specifies approximately 120 subjects while the other specifies approximately 126 subjects. At the same time, sections describing the visit schedule are correctly classified as matches because both documents specify the same sequence of visits despite minor wording differences.

These results demonstrate how the application highlights substantive changes across document versions while filtering out superficial textual variation.

theme	issue_type	description	file1_detail	file2_detail
Study design	Discrepancy	Matched: Both describe the core randomization design with the same structure: subjects randomized in a 1:1 ratio to receive Treatment A or placebo. This is the identical fundamental study design concept, differing only in subject count. Numeric difference: File 1 specifies approximately 120 subjects; File 2 excerpt 3 specifies approximately 126 subjects	The excerpt describes the overall study design specifying that approximately 120 subjects will be randomized in a 1:1 ratio to receive Treatment A or placebo.	Describes the basic randomization structure with approximately 126 subjects assigned in a 1:1 ratio to Treatment A or placebo.
Study design	Match	FILE 2 excerpt #5 describes the exact same visit schedule with identical timing windows: Screening (Day -28 to Day -1), Baseline (Day 1), Week 4 (±7 days), Week 8 (±7 days), and Week 12 (±10 days). Both excerpts focus on the scheduled visit protocol as a core component of the study design structure.	The excerpt describes the visit schedule as part of the study design, specifying 5 scheduled visits: Screening (Day -28 to Day -1), Baseline (Day 1), Week 4 (±7 days), Week 8 (±7 days), and Week 12 (±10 days).	This section outlines the scheduled visit protocol for the study, specifying the visit timeline with exact day ranges: Screening (Day -28 to Day -1), Baseline (Day 1), Week 4 (±7 days), Week 8 (±7 days), and Week 12 (±10 days).

Display 2. Document Version Control Output Showing Semantic Matches and Discrepancies Between Two Versions of a Statistical Analysis Plan

PRACTICAL IMPACT

Compared with manual document review, the proposed approach significantly reduces the time required to compare protocols, SAPs, and document revisions. Instead of scanning entire documents line by line, reviewers can focus directly on theme-level differences automatically identified across sections.

Compared with sending full documents directly to an LLM, the structured workflow improves reliability and traceability. Because each analytical unit retains its section context, the system reduces risks such as hallucinated associations or cross-section inference, while allowing reviewers to trace results back to the original document sections.

Processing documents as section-level units also improves efficiency. The approach avoids repeatedly analyzing entire documents, making model usage faster and more cost-effective. This structure also allows comparisons to scale across multiple documents and versions, supporting broader study-level review workflows.

DISCUSSIONS

Clinical study documents are typically long and heavily structured. When such documents are treated as unstructured text and submitted directly to a language model, unclear section boundaries can lead to hallucinated associations or cross-section inference. These risks become more pronounced as document size and the number of documents increase, because the model must interpret increasingly complex context.

The proposed workflow addresses this challenge by first reconstructing the document hierarchy and then performing theme-aligned analysis. Related content is identified and processed within the same thematic scope, reducing the likelihood that similar information appearing in different parts of a document will be misinterpreted. Controlling the length of each model input also helps limit hallucination. Because the workflow processes documents as structured units rather than full documents, performance remains stable even as document size or the number of documents increases.

At the same time, several limitations should be acknowledged. Clinical trial documents, whether in Word or PDF format, can vary substantially in structure and formatting. As a result, the heading detection step may require case-by-case adjustment to reliably identify section boundaries. In addition, both theme extraction and document comparison involve repeated LLM calls, which may introduce operational constraints. In enterprise settings, LLM usage is often subject to rate limits or call frequency restrictions, which can affect overall processing speed.

Despite these limitations, the approach may support a broader range of document analysis tasks beyond the examples presented in this paper. The same structure-based workflow could be applied to other regulatory documents such as clinical study reports (CSRs) or submission packages. Because the method operates on section-level units, it may also support comparisons across multiple studies or document versions, enabling more scalable review workflows in clinical development.

CONCLUSION

Reviewing and comparing clinical trial documents remains a time-consuming task, particularly when protocols, SAPs, and document revisions must be evaluated manually. Although LLMs offer strong potential for document analysis, directly applying them to long, highly structured clinical documents may produce unreliable results, including hallucination and cross-section inference, because document boundaries and hierarchical context are not preserved.

To address this problem, this paper presented a preprocessing workflow that reconstructs document structure before LLM interaction and preserves that structure throughout downstream analysis. After heading detection and hierarchy reconstruction, content is organized into section-aligned chunks, with section context retained and carried forward during theme-based analysis and annotation. This design

allows document content to be interpreted within its intended analytical scope rather than as unstructured text.

The proposed approach supports document analysis that is more efficient, more cost-effective, and more reliable than either manual review or direct whole-document LLM processing. The example applications further demonstrate how the resulting structured outputs can support practical tasks such as section-aligned document comparison, discrepancy detection, and version tracking. Because the workflow is independent of any specific programming language or language model, the main contribution of this work lies in the structure-preserving design itself and in the consistent use of document hierarchy throughout the analysis process.

Overall, the proposed methodology provides a practical and extensible foundation for applying LLMs to complex clinical trial documents while maintaining the structure, context, and traceability required for clinical development workflows.

REFERENCES

U.S. Food and Drug Administration (FDA). *Electronic Systems, Electronic Records, and Electronic Signatures in Clinical Investigations: Questions and Answers.* Guidance document. Accessed January 16, 2026. Available at https://www.fda.gov/regulatory-information/search-fda-guidance-documents/electronic-systems-electronic-records-and-electronic-signatures-clinical-investigations-questions

Gomez-Cabello, Cesar A., Srinivasagam Prabha, Syed A. Haider, Ariana Genovese, Bernardo G. Collaco, Nadia G. Wood, and Sanjay Bagaria. 2025. “Comparative Evaluation of Advanced Chunking for Retrieval-Augmented Generation in Large Language Models for Clinical Decision Support.” *Bioengineering* 12(11):1194.

Anh-Hoang, Dang, Vu Tran, and Le-Minh Nguyen. 2025. “Survey and Analysis of Hallucinations in Large Language Models: Attribution to Prompting Strategies or Model Behavior.” *Frontiers in Artificial Intelligence* 8:1622292.

Yang, Zichao, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. “Hierarchical Attention Networks for Document Classification.” In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1480–1489. San Diego, CA: Association for Computational Linguistics.

ACKNOWLEDGEMENTS

The authors would like to thank Xin Sun for reviewing the manuscript and providing valuable feedback, and Xiangdong Liu for insightful comments and support. In addition, we appreciate Xinran Wang for helpful statistical input. Finally, we especially thank Lu Zhang for her valuable help during the preparation of this manuscript.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Zun Wang
R&G US Inc
zun.wang@rgpharmaus.com
Juntao Yan
Eli Lilly and Company
juntao.yan@lilly.com