IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 15, No. 2, April 2026, pp. 1909~1918
ISSN: 2252-8938, DOI: 10.11591/ijai.v15.i2.pp1909-1918

Evaluating document chunking approaches for retrieval-augmented generation in editorial content

Erwann Lavarec, Yu Du
Department of Research and Development, Cloud is Mine, Montpellier, France

Article Info

Article history:

Received Sep 24, 2025
Revised Jan 13, 2026
Accepted Feb 6, 2026

Keywords:

Document chunking
Information retrieval
Large language model
Natural language processing
Question answering
Retrieval-augmented generation
Text segmentation

ABSTRACT

Retrieval-augmented generation (RAG) systems promise grounded answers from large language models (LLMs), yet performance depends critically on how source documents are segmented before indexing. This study investigates how pre-index chunking strategies affect both retrieval accuracy and answer quality in domain-specific scenarios. We curated a corpus on software-as-a-service (SaaS) editorial content and constructed a high-quality evaluation dataset containing 2,419 question-answer (QA) pairs generated through automated prompting and quality control. We compared four chunking approaches, including fixed-size, structure-aware recursive, semantic, and LLM-based methods. Our evaluation protocol assessed retrieval through document localization, semantic similarity, and context relevance, while generation quality was evaluated using chain-of-thought (CoT) criteria driven by judgments from LLMs. Results demonstrate that recursive chunking consistently outperforms other approaches across all metrics. Smaller chunks improve document localization, while moderately larger chunks enhance semantic alignment and generation scores. LLM-based chunking variants show competitive performance but do not exceed top recursive configurations on the dataset. These findings indicate that preserving document structure through recursive chunking is beneficial for practical RAG implementations, providing actionable guidance for chunk size selection while highlighting token-budget constraints in current long-context models.

Corresponding Author:

Yu Du
Department of Research and Development, Cloud is Mine
CA Center, 621 Rue Georges Méliès, 34000 Montpellier, France
Email: yu.du@appvizer.com

1. INTRODUCTION

While large language models (LLMs) have transformed natural language processing capabilities, they remain vulnerable to hallucination and factual drift when queries extend beyond their training data boundaries [1]. Retrieval-augmented generation (RAG) emerged as a critical solution to this limitation, systematically grounding LLM responses in external evidence by integrating dedicated retrieval mechanisms with generative models [2]–[4]. However, real-world deployments of RAGs face a fundamental challenge: they must process extensive, structurally complex documents containing diverse elements, including headings, paragraphs, lists, and tables. In these practical applications, system performance depends not only on retriever sophistication and LLM capabilities, but also on the often-overlooked process of document chunking during pre-indexing. This chunking directly determines what content becomes available to the retriever and consequently, what context reaches the LLM, making it a pivotal factor in both retrieval relevance and final answer quality. By analogy with electronic systems, where the first amplification stage

largely determines the signal quality throughout the processing chain, the authors posit that the effectiveness and robustness of the initial processing stage in a RAG pipeline play a decisive role in the overall quality of the generated responses.

Despite its fundamental importance, document chunking remains poorly understood and inadequately validated across domain-specific applications. Practitioners typically resort to generic heuristics, e.g., "use 1K characters with 5-10% overlap", while research efforts concentrate primarily on downstream components like retrieval algorithms, reranking mechanisms [5], and generator prompting strategies. This issue is particularly problematic given that chunking decisions embody a fundamental trade-off with no universal solution: smaller chunks enhance precise localization within a document but risk fragmenting coherent meaning across artificial boundaries; larger chunks preserve topical context and semantic relationships but introduce irrelevant noise and strain token budgets, compounding the long-context reliability challenges that plague modern LLMs [6]. The complexity deepens when considering the diverse methodological approaches available. Fixed-size windows, structure-aware recursive segmentation, semantic boundary detection, and LLM-driven segmentation each involve distinct trade-offs, which may interact with domain characteristics and document types in unpredictable ways. Yet comprehensive, end-to-end evaluations that systematically connect chunking strategies to both retrieval effectiveness and answer quality within specific domains remain conspicuously absent from the literature. While recent surveys and benchmarks have expanded RAG evaluation frameworks [7]–[9], actionable guidance for practitioners working with specialized professional corpora remains elusive.

This work addresses this critical gap through a comprehensive evaluation in the editorial domain that addresses the topic of software-as-a-service (SaaS). We systematically curate a high-quality dataset from two established media sources in the SaaS field and construct a rigorous question-answer (QA) evaluation set using a state-of-the-art LLM to generate fact-seeking questions directly grounded in source passages, with automated quality controls ensuring dataset integrity. Our evaluation framework introduces a multi-dimensional assessment protocol that captures both retrieval effectiveness and generation quality. We measure retrieval performance across three complementary metrics: i) document-level localization, ii) semantic alignment, and iii) pragmatic relevance. To assess end-to-end system performance, we employ LLM-judged [10], chain-of-thought (CoT) evaluation of final answer quality, providing insight into how chunking decisions ultimately impact user-facing outcomes. Within this unified experimental process, we conduct systematic comparisons across four major chunking families, including fixed-size, recursive (structure-aware), semantic boundary detection, and LLM-based approaches.

This paper delivers three key contributions to the RAG chunking literature: a meticulously curated, domain-specific QA resource for SaaS editorial content with rigorous quality controls; a holistic evaluation protocol that systematically connects chunking strategies to both retrieval effectiveness and final answer quality; and evidence-backed, actionable guidance for optimizing chunk size and segmentation approaches in RAG systems. By grounding our analysis in a realistic corpus and employing evaluation metrics that integrate both computational similarity measures and nuanced LLM assessments, this work bridges the gap between theoretical chunking research and practical deployment needs. Our findings provide practitioners with the principled, empirically validated design guidance necessary for building RAG systems that are both reliable and efficient in real-world applications.

The remainder of the paper is structured as follows. Section 2 reviews prior work and outlines the families of chunking strategies considered in this study. Section 3 describes the dataset construction and the evaluation protocol, including retrieval and generation metrics. Section 4 reports the empirical results and discusses their implications and trade-offs. Section 5 concludes with key takeaways and directions for future research.

2. RELATED WORKS AND CHUNKING STRATEGIES

The RAG approach integrates an information retrieval component with an LLM to generate answers that are grounded in external knowledge [2], [3]. As illustrated in Figure 1, a RAG system operates through three principal phases: vector database building, retrieval, and generation based on LLM. First, in the vector database building phase, a textual corpus is segmented into smaller, retrievable units using a chunking model. Each chunk is then transformed into a dense numerical representation using an embedding model. The resulting embeddings are stored in a vector database to enable efficient, similarity-based access. Second, during the retrieval phase, a user query is converted into its own embedding representation (dense vector) using the same embedding model, and a semantic search is performed within the vector database to identify the top-N most relevant chunks according to cosine similarity or another distance metric. To accelerate large-scale retrieval, modern RAG systems often rely on approximate nearest neighbor (ANN) algorithms such as hierarchical navigable small world graphs (HNSW) [11], inverted file indexes (IVF) [12], and

locality-sensitive hashing (LSH) [13]. These algorithms provide sub-linear query time while maintaining high recall. Finally, in the generation phase, the retrieved chunks are concatenated and provided as external context to the LLM, which synthesizes a final answer grounded in the retrieved evidence. This step ensures factual consistency and reduces hallucination by conditioning the model's output on verified content.

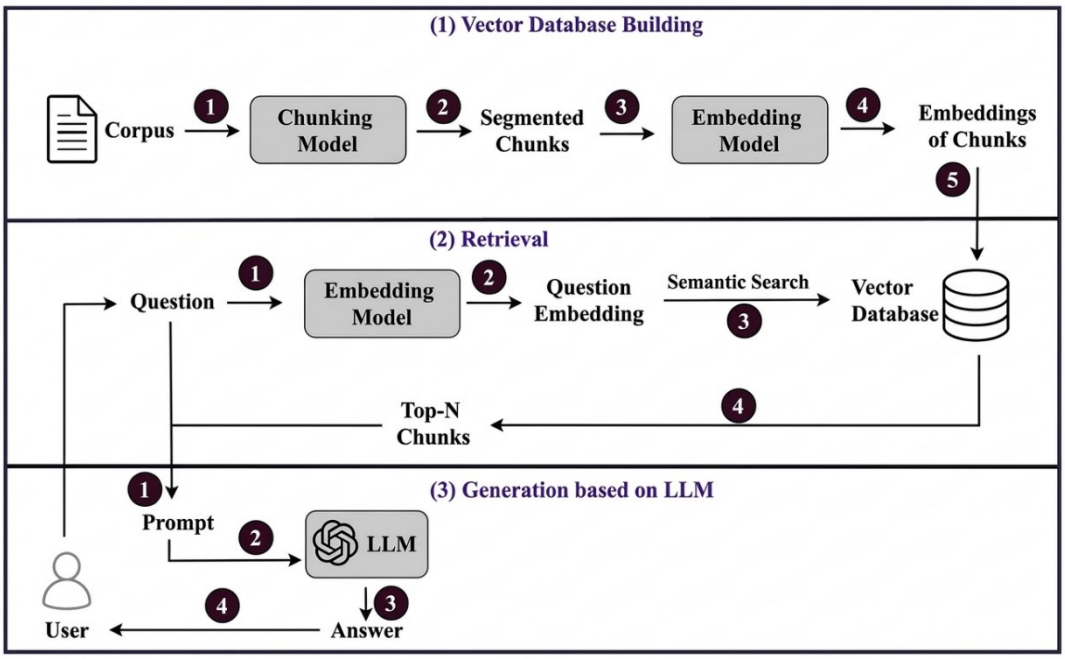

Figure 1. General workflow of a RAG system

While the quality of knowledge representation [14], the efficacy of retrieval [15]–[17], and the performance of LLMs [18] have been the focus of extensive research in the RAG field, the pre-index chunking step remains a pivotal factor. This step defines the granularity of indexed units and consequently influences both retrieval accuracy and generation quality. The selection of chunking is a critical factor in the efficacy of information retrieval. Poorly chosen chunking can result in the fragmentation of coherent information or, conversely, the overloading of the retriever with overly broad segments. In both cases, downstream performance is negatively impacted.

Document chunking strategies have evolved from simple heuristics to sophisticated LLM-driven approaches. Table 1 presents a taxonomy of the main chunking strategies, summarizing their core principles, strengths, and limitations. A detailed explanation of each chunking strategy is provided in the next paragraphs.

Table 1. Taxonomy of chunking strategies for RAG

Chunking strategy	Description	Strengths	Weaknesses
Fixed-size character chunking	Splits text into uniform segments by character count (e.g., every 1,000 characters)	Simple and fast to implement	Breaks context and ignores content boundaries
Recursive chunking	Splits by structural markers (sections, paragraphs, and lines), using a recursive hierarchy of separators	Preserves document structure, better context	Sometimes produces uneven sizes
Semantic chunking	Chunks are determined using semantic similarity, embeddings, or topic changes; maintains meaning continuity	Produces coherent, meaningful chunks that improve retrieval	Computationally intensive
LLM-based chunking	Utilizes an LLM to intelligently segment documents at semantically appropriate boundaries	Creates highly semantic and context-rich chunks	Computationally expensive

Fixed-size character chunking represents the foundational method, dividing documents into contiguous windows of predetermined character length [2]. Recursive chunking builds upon this foundation

by incorporating structural awareness—applying size constraints while preserving natural document boundaries such as headings, paragraphs, and sentence breaks. More sophisticated approaches focus on semantic coherence. Semantic chunking identifies optimal segmentation points by detecting topical shifts, typically through continuous monitoring of semantic similarity between adjacent sentences, and placing boundaries where coherence falls below a defined threshold [19]–[21]. Nevertheless, recent studies [21] question semantic chunking's effectiveness given its substantial computational overhead, which requires both embedding generation and similarity calculations for every potential boundary decision.

Given LLMs' demonstrated excellence across diverse natural language processing tasks, recent research has increasingly focused on leveraging these models directly for intelligent document chunking. LLM-based chunking approaches harness the reasoning capabilities of LLMs to make intelligent segmentation decisions. These range from straightforward prompting strategies to advanced architectures like the mixture-of-chunkers (MoC) method [22], which employs a routing mechanism to dynamically select the most appropriate granularity for different document sections, or LumberChunker [23], which employs LLMs to dynamically segment documents into semantically independent chunks.

Recent research has increasingly focused on systematic evaluation of chunking strategies across diverse contexts and applications. The OmniEval benchmark [7] established a comprehensive evaluation framework for the financial domain, systematically varying chunk sizes, overlap parameters, embedding models, and retrieval mechanisms to assess their combined impact on performance. Building on this foundation, the MoC approach [22] demonstrated that intelligently routing between multiple segmentation granularities can significantly enhance downstream RAG performance.

The field has since expanded beyond pure segmentation optimization. ChunkRAG [24] shifts the paradigm by introducing LLM-based post-retrieval filtering, focusing on refining and curating retrieved chunks rather than perfecting initial segmentation—specifically targeting the removal of irrelevant or redundant context before answer generation. Meanwhile, comparative studies [9] have examined the trade-offs between late chunking [25] and contextual retrieval [26], revealing that while contextual retrieval better preserves semantic coherence, it comes with substantial computational overhead. Late chunking offers greater efficiency but often at the expense of retrieval relevance and answer completeness.

3. EXPERIMENTS

This section provides the experimental details of our study, including the dataset construction and the evaluation protocol. Sub-section 3.1 describes the creation of a high-quality, domain-specific corpus and the generation of ground-truth QA pairs used for evaluation. Sub-section 3.2 details the evaluation framework, detailing the retrieval and generation metrics, along with the configurations used to assess the performance of various chunking strategies within the RAG pipeline.

3.1. Dataset

We assembled a domain-specific corpus by curating high-quality editorial articles on SaaS software from two established media sources: Appvizer (https://www.appvizer.com/) and Blog du Modérateur (https://www.blogdumoderateur.com/). After collecting and normalizing the raw text from the candidate web pages, we constructed a ground-truth QA dataset following a structured generation and validation process [8], [27]. Each article was first segmented into relatively large passages, which were then used as context in prompts for an LLM (GPT-4o in our case) to generate concise, fact-seeking questions paired with their corresponding answers, both strictly grounded in the source passage. To ensure quality, the resulting QA pairs were evaluated by GPT-4o itself as a judge, which rated their clarity, factual accuracy, and answerability on a five-point scale. Only pairs receiving scores above four out of five on all criteria were retained. As a result, 2,419 QA pairs, each aligned with its corresponding passage, were retained for evaluation. This pipeline yielded a scalable, high-quality QA resource tailored to the publishing industry regarding SaaS, providing a reliable basis for the systematic evaluation of chunking strategies in RAG. Figure 2 presents an example of a passage–question–answer triple. Each passage was used to generate a fact-seeking question and its corresponding answer, both grounded in the source passage

3.2. Evaluation protocol

We evaluated the candidate chunking strategies across both the retrieval and generation phases of the RAG pipeline. Our protocol combines traditional similarity-based metrics with LLM-based judgments, thereby capturing complementary aspects of performance. This comprehensive evaluation design ensures that both the quantitative precision of retrieval and the qualitative reasoning ability of the generative model are systematically assessed within a unified experimental framework.

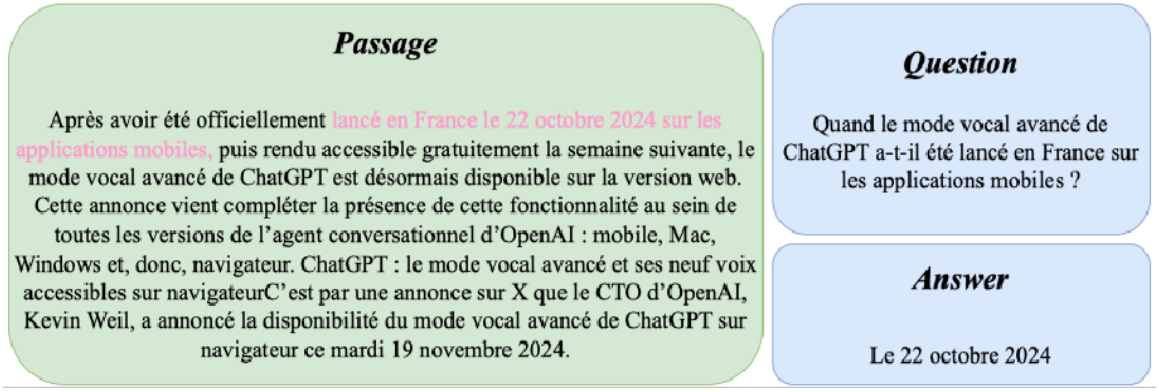

Figure 2. Illustration of a passage–question–answer triple

3.2.1. Retrieval evaluation

We assess how effectively retrieved document chunks align with the gold reference passage used to construct the corresponding QA pair. The general purpose of this evaluation is to measure the capability of identifying relevant context to the question for the LLM. Three metrics were used:

i) Simple context precision: this metric measures the proportion of retrieved chunks that originate from the same source raw document as the gold reference passage. It directly reflects the document-level localization capability of the retriever.

ii) Semantic similarity: this metric computes the cosine similarity between its embedding and the embedding of the gold passage. This metric captures semantic closeness between the retrieved context and the source passage. OpenAI's text-embedding-3-large model was used.

iii) LLM-based context precision: this metric uses an LLM evaluator to judge whether each retrieved context is relevant to answering the given question, regardless of document provenance. This metric accounts for pragmatic usefulness and can highlight cases were traditional similarity measures underestimate retrieval quality. In our experiments, GPT-4o was used as the evaluator. The implementation of this metric followed the retrieval augmented generation assessments (RAGAs) framework [28], which provides standardized tools for LLM-based RAG assessment.

3.2.2. Generation evaluation

We further evaluate the end-to-end capacity of the RAG system to produce correct and informative answers. For this, we employ GPT-4o as a judge. Using a CoT prompting strategy [29], the model is first instructed to provide structured feedback by comparing the generated answer with both the input question and the gold reference answer. It then assigns a quality score on a five-point scale, reflecting the answer's accuracy, factual correctness, and completeness. The scoring rubric, defining anchor criteria for each score from 1 to 5, is provided as follows together with the full evaluation prompt template used in our experiments to ensure transparency and reproducibility.

Prompt template used in LLM-as-a-judge for generation evaluation
###Task Description:
An instruction, a response to evaluate, a reference answer, and a score rubric representing evaluation criteria are given.
1. Write detailed feedback that assess the quality of the response strictly based on the given score rubric, not evaluating in general.
2. After writing a feedback, assign a score that is an integer between 1 and 5. You should refer to the score rubric.
3. The output format should be as follows: "Feedback: {{write your feedback for criteria}} [RESULT] {{an integer number between 1 and 5}}"
4. Please do not generate any other opening, closing, and explanations. Be sure to include [RESULT] in your output.
###The instruction:
{instruction}
###Response to evaluate:
{response}

###Reference Answer:
{reference_answer}
###Score Rubrics:
[Is the response correct, accurate, and factual based on the reference answer?]
Score 1: The response is completely incorrect, inaccurate, and/or not factual.
Score 2: The response is mostly incorrect, inaccurate, and/or not factual.
Score 3: The response is somewhat correct, accurate, and/or factual.
Score 4: The response is mostly correct, accurate, and factual.
Score 5: The response is completely correct, accurate, and factual.
###Feedback:

3.2.3. Chunker configurations

For the evaluation, we analyzed four distinct document chunking strategies. The baseline approaches consisted of fixed-size character chunking and rec.ursive chunking, both of which were evaluated at intervals of 500, 750, 1,000, 1,250, and 1,500 characters. For more context-aware segmentation, we utilized semantic chunking with a percentile-based breakpoint threshold of 0.95. Finally, we incorporated LLM-based chunking implemented via GPT-4o, testing both a standard basic approach [22] and a MoC variant [22].

4. RESULTS AND DISCUSSION

This section presents the empirical results of our study, comparing chunking strategies across both retrieval and generation metrics. The analysis highlights how different segmentation approaches influence retrieval precision, semantic similarity, and end-to-end answer quality. In addition, the discussion interprets these findings to identify trade-offs between chunk size, structural awareness, and overall RAG performance.

Figure 3 illustrates the impact of chunk size on retrieval performance for the character-based methods (fixed-size and recursive). As demonstrated in the following investigation, for the simple context precision metric, smaller chunks consistently yield higher document-level localization. Moreover, recursive chunking outperforms fixed-size windows at all scales. Semantic similarity has been shown to increase with chunk size, reaching a peak at approximately 1,250-1,500 characters. This suggests that larger segments may preserve more topical context. Finally, LLM-based context precision underscores a trade-off: recursive chunking attains optimal performance at intermediate sizes (approximately 750 characters), while fixed-size chunking maintains greater stability across the spectrum. These complementary patterns indicate that the chunk size has a direct impact on the equilibrium between fine-grained localization and broad semantic coverage.

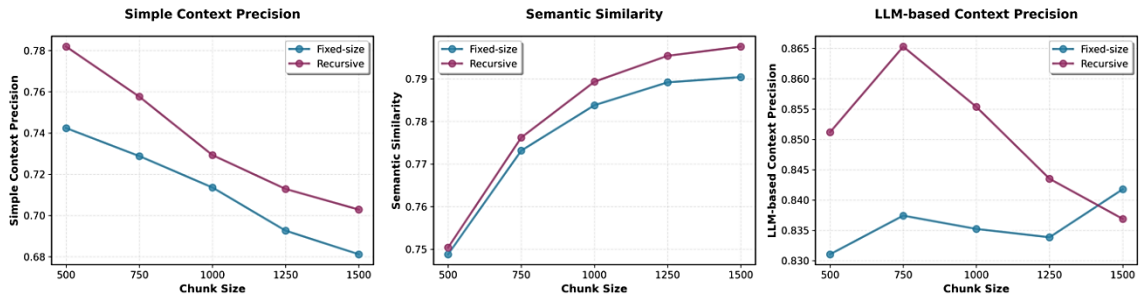

Figure 3. Performance comparison of fixed-size and recursive text chunking strategies across different chunk sizes for three retrieval evaluation metrics

Figure 4 reveals distinct performance patterns across chunking strategies when evaluated on three retrieval metrics and one generation metric. Recursive chunking emerges as the clear winner for retrieval tasks, achieving consistently superior scores across all metrics, with peak performance at intermediate to large chunk sizes (750-1,250 characters). This approach significantly outperforms both fixed-size (Figure 4(a)) and semantic chunking methods (Figure 4(b)). The LLM-based MoC variant [22] shows strong competitive performance, particularly excelling in LLM-based context precision (Figure 4(c)), though it falls just short of the best recursive configurations.

Conversely, semantic chunking exhibits a notable underperformance across all established retrieval metrics. The basic LLM-based chunker fares somewhat better, delivering moderate outcomes that generally bridge the performance gap between the baseline fixed-size methods and the more advanced MoC approaches. Together, these results highlight the distinct advantage of recursive segmentation, which strikes a critical balance between preserving document structure and maintaining semantic coherence.

The generation quality results (Figure 4(d)) reveal an even more pronounced advantage for recursive chunking. At larger chunk sizes (1,250-1,500 characters), recursive segmentation dominates with LLM-judge scores exceeding 4.3 on the five-point scale—representing the highest performance across all methods tested. While fixed-size chunking maintains steady performance, it consistently falls short of recursive peaks, and semantic chunking shows markedly weaker results. The LLM-driven approaches follow a familiar pattern: MoC outperforms the basic LLM chunker but neither variant matches the top-performing recursive configurations. These converging results across both retrieval and generation metrics highlight the effectiveness of recursive chunking, whose preservation of document structure and ability to accommodate moderately large chunk sizes provide an optimal balance.

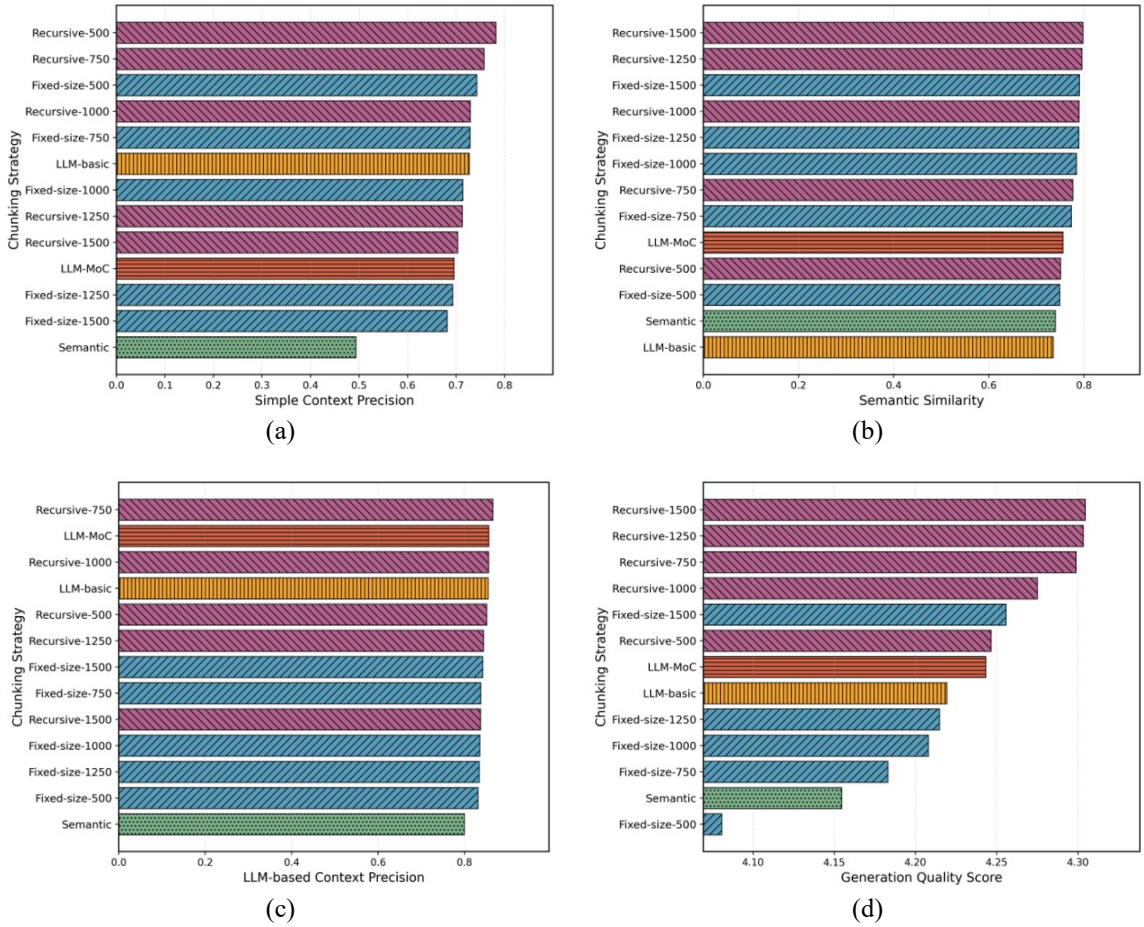

Figure 4. Comparison of chunking strategies across retrieval metrics of (a) simple context precision, (b) semantic similarity, (c) LLM-based context precision, and (d) generation quality

These findings highlight several important trade-offs for designing RAG systems in the editorial domain dealing with the subject of SaaS. First, chunk size exerts a direct influence on retrieval quality: smaller segments facilitate document-level localization, whereas moderately larger ones improve semantic alignment. Larger chunk sizes generally lead to improved generation performance. However, recent findings from Chroma's technical report show that when models are exposed to very long contexts, i.e., on the order of 10^4 tokens, their performance can degrade due to positional biases and reduced reliability, even on relatively simple tasks [6].

Overall, recursive chunking emerges as the most effective, consistently outperforming fixed-size, semantic, and LLM-driven approaches in both retrieval and generation tasks. Nevertheless, the relative underperformance of semantic and basic LLM-based chunkers suggests that sophisticated segmentation alone does not guarantee better downstream utility without careful calibration. While the present study focuses on editorial content regarding SaaS, it is hypothesized that the advantages of recursive chunking likely extend to other specialized and non-specialized editorial domains. Recent research in the field of legal document retrieval [30], [31] has demonstrated the efficacy of recursive chunking in preserving the hierarchical structure and cross-references that are critical for legal interpretation. These findings suggest that domains characterized by inherent structural organization may particularly benefit from this approach. In a similar vein, technical documentation and scientific papers—which frequently contain sections that reference definitions or concepts introduced elsewhere—have been demonstrated to exhibit optimal performance when utilizing recursive and hierarchical chunking strategies. Consequently, we hypothesize that the advantages of recursive chunking will transfer to a broader range of editorial domains when reliable structural or formatting signals are available. Future research should empirically validate recursive chunking's effectiveness across these diverse domains to establish its broader applicability and identify domain-specific optimization requirements.

Beyond cross-domain applicability, the findings of this study also connect to broader research efforts in RAG optimization and long-context modeling. Recent evidence shows that LLMs exhibit context decay and fail to uniformly attend to information distributed across long input sequences [32]. This limitation highlights the importance of supplying models with coherent, well-structured context segments rather than long, noisy concatenations of text. By demonstrating that recursive chunking better preserves document structure while maintaining manageable segment lengths, our results contribute to this wider line of research seeking to improve the robustness, factual consistency, and efficiency of RAG systems operating under long-context constraints.

5. CONCLUSION

This study systematically investigated the impact of document chunking strategies on RAG performance within the editorial domain regarding SaaS. The results demonstrate that pre-indexing segmentation decisions strongly influence both retrieval effectiveness and answer quality, with recursive chunking using moderately large segments (750-1,250 characters) consistently yielding the best balance between document localization and semantic coherence. These findings highlight that structural document awareness remains essential, even when compared with more sophisticated semantic-aware or LLM-driven chunking methods, and that effective chunking must balance retrieval precision with token budget and long-context reliability constraints. Looking ahead, future research should generalize these observations across other domains such as legal or scientific texts, explore adaptive chunking combined with hybrid search strategies mixing several approaches, dynamic retrieval or reranking strategies, and investigate how segmentation must evolve alongside emerging long-context architectures. By providing a domain-specific benchmark, a comprehensive evaluation framework, and empirical guidance on optimal chunking design, this work advances both the theoretical understanding and practical deployment of reliable, efficient RAG systems.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Erwann Lavarec	✓	✓		✓			✓		✓	✓		✓	✓	
Yu Du	✓	✓	✓	✓	✓		✓	✓		✓	✓			

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that they have no conflicts of interest related to this work.

DATA AVAILABILITY

The dataset supporting the findings of this study is publicly available in Zenodo at https://doi.org/10.5281/zenodo.17423337. The dataset is distributed under the Creative Commons Attribution 4.0 International (CC BY 4.0) license, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. A mirrored version of the dataset is also hosted on Hugging Face at https://huggingface.co/datasets/appvizer/qa-dataset-for-saas to facilitate access for the research community.

REFERENCES

[1] L. Huang *et al.*, "A survey on hallucination in large language models: principles, taxonomy, challenges, and open questions," *ACM Transactions on Information Systems*, vol. 43, no. 2, pp. 1–55, 2025, doi: 10.1145/3703155.

[2] P. Lewis *et al.*, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020, pp. 9459–9474, doi: 10.5555/3495724.3496517.

[3] W. Fan *et al.*, "A survey on RAG meeting LLMs: towards retrieval-augmented large language models," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 6491–6501, doi: 10.1145/3637528.3671470.

[4] Y. Gao *et al.*, "Retrieval-augmented generation for large language models: a survey," 2023, *arXiv:2312.10997*.

[5] N. Ampazis, "Improving RAG quality for large language models with topic-enhanced reranking," in *Artificial Intelligence Applications and Innovations*, Cham, Switzerland: Springer, 2024, pp. 74–87. doi: 10.1007/978-3-031-63215-0_6.

[6] K. Hong, A. Troynikov, and J. Huber, "Context rot: how increasing input tokens impacts LLM performance," *Chroma Research*. 2025, Accessed: Sep. 02, 2025, [Online]. Available: https://research.trychroma.com/context-rot

[7] S. Wang, J. Tan, Z. Dou, and J.-R. Wen, "OmniEval: an omnidirectional and automatic RAG evaluation benchmark in financial domain," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025, pp. 5737–5762, doi: 10.18653/v1/2025.emnlp-main.292.

[8] K. Zhu *et al.*, "RAGEval: scenario specific RAG evaluation dataset generation framework," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025, pp. 8520–8544, doi: 10.18653/v1/2025.acl-long.418.

[9] C. Merola and J. Singh, "Reconstructing context: evaluating advanced chunking strategies for retrieval-augmented generation," in *Knowledge-Enhanced Information Retrieval*, Cham, Switzerland: Springer, 2026, pp. 3–18, doi: 10.1007/978-3-032-02899-0_1.

[10] L. Zheng *et al.*, "Judging LLM-as-a-judge with MT-bench and chatbot arena," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, 2023, vol. 36, pp. 46595–46623, doi: 10.5555/3666122.3668142.

[11] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 4, pp. 824–836, 2020, doi: 10.1109/TPAMI.2018.2889473.

[12] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 117–128, 2011, doi: 10.1109/TPAMI.2010.57.

[13] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proceedings of the 25th International Conference on Very Large Data Bases*, 1999, pp. 518–529, doi: 10.5555/645925.671516.

[14] Z. Xu *et al.*, "Retrieval-augmented generation with knowledge graphs for customer service question answering," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024, pp. 2905–2909, doi: 10.1145/3626772.3661370.

[15] S. Li, L. Stenzel, C. Eickhoff, and S. Ali Bahrainian, "Enhancing retrieval-augmented generation: a study of best practices," in *Proceedings of the 31st International Conference on Computational Linguistics*, 2025, pp. 6705–6717.

[16] Y. Yu *et al.*, "RankRAG: unifying context ranking with retrieval-augmented generation in LLMs," in *38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024, pp. 121156–121184. doi: 10.52202/079017-3850.

[17] L. Gao, X. Ma, J. Lin, and J. Callan, "Precise zero-shot dense retrieval without relevance labels," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023, pp. 1762–1777, doi: 10.18653/v1/2023.acl-long.99.

[18] M. A. C. Blandón, J. Talur, B. Charron, D. Liu, S. Mansour, and M. Federico, "MEMERAG: a multilingual end-to-end meta-evaluation benchmark for retrieval augmented generation," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025, pp. 22577–22595, doi: 10.18653/v1/2025.acl-long.1101.

[19] LangChain, "LangChain overview," *LangChain Documents*, Accessed: Sep. 02, 2025. [Online]. Available: https://python.langchain.com/docs/how_to/semantic-chunker/

[20] LlamaIndex, "Semantic chunker," *LlamaIndex*, 2024, Accessed: Sep. 02, 2025. [Online]. Available: https://developers.llamaindex.ai/python/examples/node_parsers/semantic_chunking/

[21] R. Qu, R. Tu, and F. S. Bao, "Is semantic chunking worth the computational cost?" in *Findings of the Association for Computational Linguistics: NAACL 2025*, 2025, pp. 2155–2177, doi: 10.18653/v1/2025.findings-naacl.114.

[22] J. Zhao *et al.*, "MoC: mixtures of text chunking learners for retrieval-augmented generation system," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025, pp. 5172–5189, doi: 10.18653/v1/2025.acl-long.258.

[23] A. V. Duarte, J. D. Marques, M. Graça, M. Freire, L. Li, and A. L. Oliveira, "LumberChunker: long-form narrative document segmentation," in *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024, pp. 6473–6486, doi: 10.18653/v1/2024.findings-emnlp.377.

[24] I. S. Singh *et al.*, "ChunkRAG: novel LLM-chunk filtering method for RAG systems," 2024, *arXiv:2410.19572*.

[25] M. Günther, I. Mohr, D. J. Williams, B. Wang, and H. Xiao, "Late chunking: contextual chunk embeddings using long-context embedding models," 2025, *arXiv:2409.04701*.

[26] Anthropic, "Introducing contextual retrieval," *Anthropic*. 2024, Accessed: Sep. 02, 2025. [Online]. Available: https://www.anthropic.com/news/contextual-retrieval

[27] S. Filice, G. Horowitz, D. Carmel, Z. Karnin, L. L.-Eytan, and Y. Maarek, "Generating Q&A benchmarks for RAG evaluation in enterprise settings," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025, pp. 469–484, doi: 10.18653/v1/2025.acl-industry.33.
[28] S. Es, J. James, L. E. Anke, and S. Schockaert, "RAGAs: automated evaluation of retrieval augmented generation," in *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, 2024, pp. 150–158, doi: 10.18653/v1/2024.eacl-demo.16.
[29] J. Wei *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2022, pp. 24824–24837, doi: 10.5555/3600270.3602070.
[30] A. F. Ferraris, D. Audrito, G. Siragusa, and A. Piovano, "Legal chunking: evaluating methods for effective legal text retrieval," in *Legal Knowledge and Information System*, IOS Press, 2024, pp. 275–281, doi: 10.3233/FAIA241255.
[31] N. Pipitone and G. H. Alami, "LegalBench-RAG: a benchmark for retrieval-augmented generation in the legal domain," 2024, *arXiv:2408.10343*.
[32] N. F. Liu *et al.*, "Lost in the middle: how language models use long contexts," *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 157–173, 2024, doi: 10.1162/tacl_a_00638.

BIOGRAPHIES OF AUTHORS

Erwann Lavarec holds a Ph.D. in robotics from LIRMM, University of Montpellier (2001), where his dissertation focused on 3D motion estimation using a camera and proprioceptive sensors. He also earned an executive MBA from Montpellier Business School (2013). He is currently chief technology officer at Appvizer (Softonic group) and previously founded and led Wany Robotics. His interests span mobile-robot localization, particle filters/Monte-Carlo methods, 3D vision, and the engineering of LLMs and agentic AI systems such as RAG, semantic reasoning, and multi-agent architecture. He is also listed as an inventor on robotics and AI patents. He won the national competition for innovative technology company creation organized by the French Ministry of Research in 1999 and 2000. He has published research papers in international conferences including IEEE ICRA, on kinematic and dynamic modeling of multi-rotor aircraft systems. He can be contacted at email: erwann.lavarec@appvizer.com.

Yu Du received his Ph.D. in Computer Science from IMT Mines Alès in 2021 with the dissertation "Des données aux connaissances: vers des recommandations plus pertinentes, diversifiées et transparentes." He also holds an M.Sc. in Computer Science (data, knowledge and natural language processing) from the University of Montpellier (2017) and a B.Sc. in Computer Science from Université Clermont Auvergne (2014). He is currently an AI research engineer and data scientist at Appvizer (Softonic group). His research interests include machine learning, recommender systems, knowledge engineering, knowledge graphs, natural language processing, information retrieval, and agentic AI. Prior to his current position, he worked as a research engineer at Cirad, developing semantic web tools for information retrieval and data analysis. He has published several papers in international journals including Information Processing and Management, Knowledge-Based Systems, and PLoS ONE, with notable contributions to collaborative filtering, recommendation diversity, and explainable AI. He can be contacted at email: du.yu.1411@gmail.com or yu.du@appvizer.com.