Granularity Reciprocal Rank Fusion: Fusing Multiple Chunk Granularities for Robust Retrieval-Augmented Generation

Jonatas Novais Tomazini[1], **Salvador Ludovico Paranhos**[1], **Celso Camilo Junior**[1], **Sávio Teles de Oliveira**[1]

[1]Instituto de Informática – Universidade Federal de Goiás (UFG)
Goiânia – GO – Brazil

tomazini@discente.ufg.br, salvadorludovico@egresso.ufg.br
{celsocamilo,savioteles}@ufg.br

Abstract. *Retrieval-Augmented Generation (RAG) systems depend heavily on how textual data is segmented and retrieved. We introduce Granularity Reciprocal Rank Fusion (GRRF), a novel approach for optimizing context selection in RAG pipelines by leveraging multiple levels of text granularity. GRRF indexes the same corpus under different chunking granularities and reranks the retrieved passages using a supervised cross-encoder. The results are then fused through the Reciprocal Rank Fusion (RRF) technique, enabling robust selection of the most informative content. Experiments on legal, news, and audit corpora demonstrate that GRRF's central, model-agnostic finding is that no single chunking granularity is optimal across domains—smaller chunks win in legal and news, and larger chunks in audit—a result read directly from the baselines and therefore independent of any reranking. GRRF is designed around this finding: instead of committing to one fragile granularity, it fuses evidence from all of them, and in our experiments it consistently surpasses strong single-pass baselines (up to 5.7% relative gain in context quality). These findings highlight the effectiveness of granularity-aware fusion for enhancing both precision and robustness in retrieval for large language models.*

1. Introduction

The *Retrieval-Augmented Generation* (RAG) approach has established itself as an efficient solution in natural language processing systems by integrating information retrieval mechanisms with generative language models [Lewis et al. 2020]. Because language models operate under a bounded context window, RAG pipelines first segment the corpus into smaller retrievable units, called *chunks*, and assemble the model input from those judged most relevant to the query. As RAG is increasingly deployed in high-stakes domains—such as law and public administration—where inaccurate information can lead to critical decisions, ensuring that retrieved content is not only relevant but also accurate and trustworthy becomes a strict requirement.

In this context, this work proposes *Granularity Reciprocal Rank Fusion* (GRRF), which applies multiple chunking methods to the same corpus, reranks the passages retrieved from each method with a supervised cross-encoder, and combines the results through *Reciprocal Rank Fusion* (RRF) [Cormack et al. 2009]. It was evaluated on case law, public audits, and news media. Our primary contribution is an empirical finding rather than a performance claim: no fixed chunking granularity dominates across heterogeneous domains (smaller chunks win in legal and news, and larger chunks win in audit).

Because this is read directly from the baseline results, it is independent of the reranking stage and shows that committing to a single granularity up front is fragile. GRRF is the natural response: it fuses evidence from all granularities instead of selecting one.

2. Related Work

This section reviews the components of RAG pipelines most relevant to our proposal and positions GRRF with respect to prior work on chunking, reranking, and retrieval fusion.

2.1. Chunking Strategies

The way a corpus is segmented into chunks is a critical design choice in RAG [Bhat et al. 2025]. Traditional strategies use fixed-length token windows, with or without overlap, while structure-aware strategies exploit document hierarchy (sections and subsections) to preserve semantic coherence [Han 2025, Schwaber-Cohen and Patel 2025]. More recent approaches address the limitations of operating at a *single* granularity. RAPTOR [Sarthi et al. 2024] builds a hierarchical tree bottom-up: chunk embeddings are projected onto a lower-dimensional space with Uniform Manifold Approximation and Projection (UMAP) and grouped by a Gaussian Mixture Model, which assigns a chunk to more than one cluster, and each cluster is condensed by a large language model (LLM), `gpt-3.5-turbo` in the original work, into a summary that becomes a parent node. Retrieval either walks the tree layer by layer or flattens it into a single pool of nodes, the latter reported as the stronger variant. The tree thus buys multiple levels of abstraction at the price of synthetic text, whose fidelity depends on the summarizer. Mix-of-Granularity [Zhong et al. 2025] keeps the original text but routes each query: a supervised router, trained with soft labels derived from how well each granularity answers a training query, predicts which of a predefined set of granularities to read from, so the system still commits to one level per query. Its graph variant preprocesses documents as graphs to reach snippets that are far apart in the source. Graph-based indexing [Chen et al. 2025] instead encodes relational structure in a two-layer index, a knowledge-graph layer over entities and a collaborative document layer over neighboring passages, so that evidence scattered across documents can be reached through graph associations. A parallel line of work segments documents hierarchically or semantically before retrieval: HiChunk [Lu et al. 2025] fine-tunes an LLM to predict where each level of a document's structure begins and ends, and pairs these multi-level boundaries with an auto-merge step that joins sibling chunks at retrieval time; TreeRAG [Tao et al. 2025] stores long documents as a tree and queries it in both directions, from root to leaves and from leaves back to the root, to keep related knowledge points together; hierarchical text-segmentation chunking [Nguyen et al. 2024] groups semantically coherent segments, and, in Brazilian-Portuguese retrieval, Toresan et al. [Toresan et al. 2025] propose a dual semantic chunker. GRRF differs from all these approaches in that it neither generates synthetic content nor commits each query to a single (structure- or semantics-aware) segmentation: it retrieves over multiple granularities in parallel and lets fusion combine their evidence.

2.2. Reranking

Reranking refines the alignment between a query and candidate chunks by scoring query–chunk pairs jointly. Cross-encoder architectures, such as the BGE-reranker family [Chen et al. 2024], are particularly effective for this task, at the cost of higher latency

compared to bi-encoder retrieval. Reranking is typically applied to a single retrieved list; in GRRF, the cross-encoder scores every query–chunk pair of each granularity-specific list separately, and each list is reordered by its own scores before fusion, so that fusion operates on already-refined rankings. Scores are never compared across lists: they only determine positions within the list that produced them.

2.3. Retrieval Fusion

Reciprocal Rank Fusion (RRF) [Cormack et al. 2009] is a rank-based aggregation method that combines multiple ranked lists without requiring score normalization. Each item is scored by $1/(k + r)$, where r is its position in a given list and k a smoothing constant, and the scores it obtains across lists are summed. Because only positions enter the formula, lists produced by different models or over different granularities—whose raw scores are not on a comparable scale, can be merged directly. In RAG, Rackauckas [Rackauckas 2024] introduced RAG-Fusion, which applies RRF to lists obtained from *multiple reformulated queries* issued against the same index. This varies the retrieval signal on the query side, but its effectiveness depends on the relevance of the generated query variants. GRRF instead varies the retrieval signal on the *indexing side*, fusing rankings produced over the same query but across multiple chunking granularities, making the two approaches orthogonal. To the best of our knowledge, no prior work fuses rankings produced over the same query across multiple chunking granularities of the same corpus, with a supervised cross-encoder inserted between retrieval and fusion.

3. Granularity Reciprocal Rank Fusion

A conventional RAG retriever measures similarity between the user query and stored chunks with embeddings, often through the Dual Encoder Architecture [Dong et al. 2022], and returns the most relevant ones to compose the context injected into the prompt.

The Granularity Reciprocal Rank Fusion (GRRF) method is structured in two main stages: document indexing and information retrieval. The first stage involves preparing and storing chunks with different levels of granularity in a vector database (Vector DB). The second stage uses a robust approach to retrieve and fuse these chunks in order to maximize the relevance and quality of the information retrieved in response to user queries. The full implementation used in our experiments is available in the source code repository[1].

3.1. Indexing Stage

Before executing GRRF, the chunking methods used to index the knowledge base must be defined. The choice of chunking methods reflects two complementary criteria. First, we adopt fixed-size chunks at exponentially increasing scales (512, 1024, and 2048 tokens) rather than arbitrary intermediate values. This range covers the trade-off commonly reported in the RAG literature [Lewis et al. 2020, Karpukhin et al. 2020]: smaller chunks favor local precision and are well suited to factual queries, while larger chunks preserve broader context at the cost of introducing less relevant content. Using clearly separated scales rather than fine-grained variations of the same range (512, 640, 768, and so on)

[1]Source code available at: `https://github.com/salvadorludovico/GRRF-RAG`

makes the contribution of each granularity to the fusion step interpretable and avoids overfitting to a particular corpus. Second, we complement these fixed-size strategies with a hierarchical method that segments documents by their native structural markers, producing variable-length chunks that preserve semantic boundaries. The method splits recursively, trying section headings first and falling back to paragraph and sentence boundaries where headings are absent, and emits chunks at more than one level, so that a section and the paragraphs inside it are both indexed. It therefore assumes documents whose layout survives text extraction as explicit markers, such as headings or numbered articles. This holds for the legal decisions and audit reports used here, each organized in named parts. On flat prose without such markers, as in the news corpus, the method degenerates to paragraph and sentence splitting and behaves like a fixed-size strategy with variable-length units. Figure 1 illustrates this process: one document enters the pipeline and leaves it as four independent sets of chunks, one per method, all indexed in the same store. In an audit report, a finding and the recommendation derived from it may sit two thousand tokens apart. At 512 tokens the finding is isolated from its recommendation, and a query about the recommendation retrieves a chunk that does not state what motivated it; at 2048 both fall in the same chunk, at the price of carrying unrelated text from the section around them. Neither cut is correct for every query, which is what motivates indexing the document under all of them.

Despite the cost of embedding multiple granularities, the additional storage is modest in optimized vector databases such as pgvector [Kane 2024], whose Hierarchical Navigable Small World (HNSW) [Malkov and Yashunin 2020] and inverted-file (IVF-Flat) indices—the latter a flat variant of the inverted file index of [Jégou et al. 2011]—offer fast similarity search with favorable scalability and moderate storage demands.

Because the hierarchical strategy aligns boundaries with structural divisions such as the *ementa* and the ruling in legal decisions, the four indices provide complementary views of the same corpus. Each resulting chunk is converted into embeddings by BGE-M3 [Chen et al. 2024], a multilingual model that produces 1024-dimensional dense vectors and covers Portuguese, and stored in the vector database (Vector DB), which underpins the retrieval stage of GRRF. The same embedding model is used for every granularity and for the query, so that all four indices and the query share one vector space and the comparison across granularities isolates the effect of segmentation rather than of the encoder.

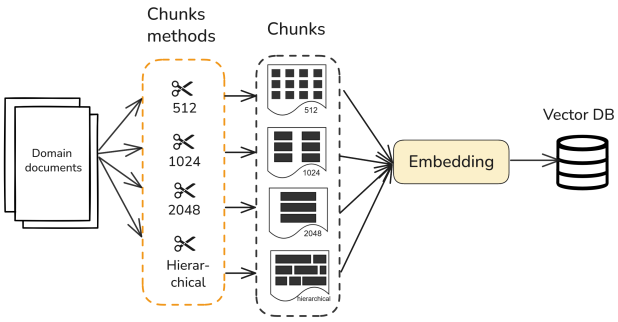


Figure 1. Domain documents are split using multiple chunking methods, converted into embeddings, and stored in a vector database.

3.2. Retriever Stage

The operation of the GRRF is illustrated in Figure 2. The process begins when the user submits a query, which is then converted into an embedding. This embedding is processed by the GRRF module, which contains the specific retrieval logic of the method, which Figure 2 expands to detail its internal operation. Simultaneously, the system accesses precomputed embeddings of chunks stored in the database. In the figure, the query is encoded once, the Top-k Retriever queries the four indices in parallel and returns four candidate lists, the reranker rescores each list on its own, and RRF merges the reordered lists into the single ranking handed to the generator.

Next, the Top-k Retriever module selects the most relevant chunks at each level of granularity (512, 1024, and 2048 tokens, plus the hierarchical strategy). For every granularity, the top-k chunks most similar to the query embedding are retrieved by cosine similarity, with $k = 20$ applied uniformly across all four indices. Cosine similarity is the measure the embedding model was trained under, and it disregards vector magnitude, which varies with chunk length and would otherwise bias retrieval towards a particular granularity, a concern specific to a multi-granularity index. This value was chosen to provide a sufficiently broad candidate pool for the subsequent reranking stage to discriminate fine-grained relevance differences, while keeping the cross-encoder cost bounded. The 80 candidates obtained across the four granularities then undergo reranking using the BGE-reranker-v2-m3 cross-encoder [Chen et al. 2024], which assigns each chunk a refined relevance score with respect to the original query. Using these scores, the four reranked lists are merged through Reciprocal Rank Fusion (RRF), whose formula is expressed as:

$$\mathrm{RRF}(d) = \sum_{i \in G} \frac{1}{c + r_i(d)} \tag{1}$$

where G is the set of index granularities (in our setup, $|G| = 4$), $r_i(d)$ denotes the position (rank) of chunk d in the ranking generated by granularity $i \in G$ after reranking, and c is a smoothing constant that limits the disproportionate influence of extreme positions. We set $c = 60$, the value originally proposed by Cormack et al. [Cormack et al. 2009] and adopted as a de facto default in the information retrieval literature.

Since chunks produced by different granularities are physically distinct objects, each chunk d appears in exactly one granularity-specific ranking $r_{i^*}(d)$, and $\frac{1}{c+r_i(d)}$ is taken as zero for $i \neq i^*$. Consequently, the summation in Eq. (1) reduces to a single non-zero term per chunk, and RRF acts here as a rank-aware interleaving of the four reranked lists rather than as a consensus operator over a shared item set. The constant $c = 60$ dampens the influence of extreme positions within each per-granularity ranking. To ensure proper alignment during fusion, each chunk receives a unique identifier prior to reranking. Finally, the chunks with the highest RRF scores, referred to as the *best chunks*, are selected and used to compose the final context, which is then injected into the prompt of the language model (LLM). Because smaller chunks may be nested within larger ones, the interleaved context can include overlapping passages; a greedy post-fusion deduplication at prompt-assembly time addresses this, and we leave its evaluation as future work.

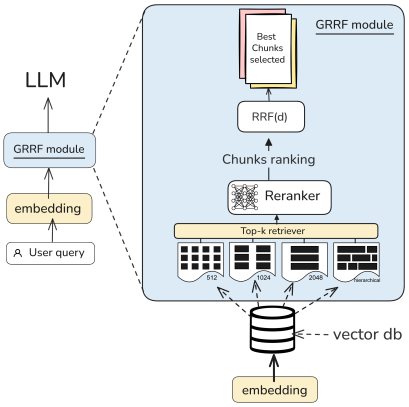


Figure 2. GRRF retrieval pipeline. A user query is embedded and matched against chunk embeddings from multiple granularities. Retrieved chunks are reranked, fused via RRF, and the top results are selected to compose the final context for the LLM.

4. Experimental Methodology

In this section, we detail the methodology employed to evaluate the GRRF chunk selection approach. We present the datasets used to apply and validate GRRF, describe the configuration of the experimental environment, and outline the set of metrics adopted to support our analysis. Throughout the evaluation, GRRF is compared against four *naive baselines*: the same pipeline restricted to a single granularity, namely fixed-size chunks of 512, 1024, and 2048 tokens, and the hierarchical strategy. They act as controls that isolate the effect of fusion, since every other component of the pipeline is held constant.

4.1. Datasets

In order to highlight the generalization and robustness of the conclusions, we used three datasets from distinct domains, whose main characteristics are summarized in Table 1. The documents were used to generate queries and indexed in a vector database, forming the foundation for the retrieval engine.

All three corpora are public. The legal corpus consists of rulings of the Brazilian Superior Court of Justice (STJ), each comprising a summary (*ementa*), the ruling text (*acórdão*), and its legal thesis (*tese*), with judgment dates between 2005 and 2025. The news corpus is a subset of the publicly available *News of the Brazilian Newspaper* dataset [Santana 2017], restricted to articles published by *Folha de S. Paulo* between January and September 2017 across sections such as politics, economy, world, and daily life. The audit corpus was retrieved from the public portal of the Goiás State Audit Court (TCM-GO) [Tribunal de Contas dos Municípios do Estado de Goiás 2024] and comprises operational audit and monitoring reports on municipalities of the state of Goiás, published between 2018 and 2024; these documents were originally distributed as PDF files and converted to text before indexing.

For each dataset, an evaluation set of 500 instances was constructed. In the legal and news domains, 500 documents were randomly sampled from the indexed corpus and one query was generated per document. In the audit domain, the corpus contains a smaller number of long and structurally rich reports, from which multiple distinct queries can be elicited per document; we therefore generated multiple queries per audit document until reaching 500 evaluation instances. Each instance is a tuple composed of: (i) a query,

Table 1. Overview and statistics of the datasets. "Indexed" = documents loaded into the vector database; "Sampled" = subset used to generate evaluation queries.

Data Source	Indexed	Sampled	Queries
STJ Jurisprudence	28,000	500	500
Folha de S. Paulo	20,000	500	500
TCM-GO Audits	200	200	500

(ii) a reference answer (golden answer), which represents the ideal and factually correct answer, and (iii) a reference context (ground-truth context), which is a literal passage extracted from the source document containing the information required to answer the query.

The performance evaluation of the strategies was conducted using three metrics derived from the Retrieval Augmented Generation Assessment (RAGAS) framework [Es et al. 2024]. These metrics were selected and weighted to emphasize the most critical aspects of retrieval quality. The final score assigned to each answer, $Score(\text{WA})$, is computed as a weighted average of the individual metric scores.

The *Context Recall* metric (weight: 0.35) captures the system's ability to retrieve all relevant chunks from the reference context. The *Context Precision* metric (weight: 0.35) assesses the proportion of relevant information within the retrieved content. Lastly, the *Faithfulness* metric (weight: 0.30) measures whether the model's generated response is factually grounded in the retrieved context. The weights (0.35, 0.35, 0.30) reflect our prioritization of retrieval quality (Context Recall and Context Precision) over generation faithfulness, since the central contribution of GRRF is on the retrieval side of the pipeline. We emphasize that these weights are adopted only for reporting a single aggregated score in Table 2 and have no effect on the conclusions drawn in this work: GRRF outperforms the naive baselines on each of the three RAGAS metrics individually in all three domains, which makes the comparison robust to any non-degenerate choice of positive weights. Per-metric scores are available in the source code repository for full transparency.

4.2. Query Generation Protocol

The evaluation tuples (query, golden answer, and ground-truth context) were generated synthetically using Gemini 2.0 Flash via the Vertex AI API, since no public Portuguese benchmark covers the three domains studied here with the required structure. The protocol was designed to ensure traceability between each query and its supporting evidence.

The generator input was restricted, per domain, to the structural fields that concentrate informative content—in law, identification metadata with the *ementa*, *acórdão*, and *tese*; in audit, the document identifier and the report body; in news, the headline, date, category, and article text—rather than arbitrary chunks, so that queries reflect realistic information needs. Prompts were further tailored per domain through a specialist persona and a list of question types to target or avoid, so as to exclude trivial or headline-answerable queries. The full prompt templates are available in the same source code repository[1].

The structured output, parsed via a schema-based parser, includes the query, a reference answer grounded exclusively in the input document, a literal reference passage of at least 20 words extracted from the document, a domain-specific tag, and a set of

keywords. The literal passage serves as the ground-truth context for retrieval evaluation, guaranteeing by construction that the supporting evidence is present in the indexed corpus, a prerequisite for metrics such as Context Recall and Context Precision.

Exhaustive manual annotation would be prohibitive for 1,500 tuples. Instead, the authors conducted a sample-based review per domain, inspecting whether each query was answerable from the source document, whether the reference answer was factually consistent, and whether the reference passage corresponded to a literal excerpt. Tuples that failed this inspection were regenerated. All final tuples, including source-document metadata, are stored as structured JSON in the repository. Although the protocol is centered on synthetic queries, the methodology also accepts real questions from domain experts when available, which we treat as a direction for future work. We acknowledge that synthetic generation introduces biases inherent to the generator model; this limitation is discussed in Section 6.

4.3. Environment and Additional Tools

The implementation of our experimental pipeline was developed in Python, supported by the LangChain library [LangChain 2024] for orchestration and Hugging Face Transformers for model access. The evaluation of the strategies was conducted using the RAGAS framework, with Gemini 2.0 Flash serving both as a query generator and evaluator. Experiments were executed on an NVIDIA DGX workstation equipped with two V100 (32 GB) GPUs. Document text extraction was carried out using the Docling tool [Auer et al. 2024], followed by preprocessing with `Pandas` for data cleaning and segmentation. The vector index was implemented via pgvector [Kane 2024], a PostgreSQL extension for similarity search over dense embeddings.

The reranking and evaluator models were chosen for multilingual capability, public benchmarking, and reproducibility. For reranking we adopted BGE-reranker-v2-m3 [Chen et al. 2024],[2] a lightweight multilingual cross-encoder (100+ languages, including Portuguese) widely used in recent RAG literature for its quality/cost trade-off [Chen et al. 2024]. As generator and evaluator within RAGAS we used Gemini 2.0 Flash, for its native Portuguese support and a cost profile compatible with large-scale evaluation (1,500 queries, with multiple metrics each), while sparing GPU memory on the retrieval hardware. We did not perform a controlled comparison among reranker or evaluator models; alternatives (e.g., `bge-reranker-v2-gemma` or GPT-4o-class evaluators) are left to future work.

5. Results and Discussion

5.1. GRRF Performance Compared to Naive Baselines

Table 2 reports the weighted-average score and average response time for every retrieval strategy across the three domains. The most important observation from Table 2 is that no fixed granularity is best across domains: 512 tokens wins in legal and news, while 2048 wins in audit. This pattern holds among the single-pass baselines themselves, so it does not depend on the reranking stage; picking a single chunk size up front is therefore risky and corpus-dependent. GRRF is built to avoid this choice by fusing rankings from

[2]Distributed on the Hugging Face Hub as `BAAI/bge-reranker-v2-m3`: `https://huggingface.co/BAAI/bge-reranker-v2-m3`

all granularities, and it achieves the highest score in all three domains, outperforming both the best naive configuration per domain and the per-domain average of the naive baselines. It improves the best naive baseline from 0.911 to 0.926 in legal (+1.6%), from 0.897 to 0.948 in news (+5.7%), and—where the strongest naive shifts to `naive_2048`, reflecting longer and more uneven documents—from 0.823 to 0.850 in audit (+3.3%). GRRF's standard deviations are comparable to or lower than the baselines', indicating stability across queries.

Table 2. Scores (weighted average) and response times (s) per strategy and domain. Bold = best per domain; response time is the average latency per query.

	Jurisprudence		**News**		**Audit**	
Strategy	Score	Time (s)	Score	Time (s)	Score	Time (s)
naive_512	0.911 ± 0.103	0.725	0.897 ± 0.116	0.512	0.744 ± 0.241	0.157
naive_1024	0.897 ± 0.122	0.372	0.888 ± 0.135	0.276	0.778 ± 0.238	0.084
naive_2048	0.870 ± 0.133	0.270	0.894 ± 0.143	0.207	0.823 ± 0.202	0.063
naive_hierarchical	0.878 ± 0.124	0.666	0.879 ± 0.133	0.347	0.771 ± 0.258	0.074
GRRF	$\mathbf{0.926 \pm 0.103}$	3.461	$\mathbf{0.948 \pm 0.095}$	2.878	$\mathbf{0.850 \pm 0.191}$	1.683

5.2. Computational Cost and Response Time Analysis

The naive strategies are the fastest, since they perform a single retrieval pass with a lighter model. GRRF, although the most accurate, incurs the highest latency: it runs multiple retrievals and, above all, relies on a cross-encoder, which scores the query and the chunk jointly instead of comparing precomputed vectors.

The trade-off is deliberate. GRRF targets domains where a response built from incomplete context can lead to legal liabilities or incorrect administrative decisions, and where added latency buys factual alignment. The cost is concrete and we report it rather than treating it as negligible: in the legal domain, GRRF raises the average response time from 0.725 s for the best naive baseline to 3.461 s, an increase driven primarily by the cross-encoder reranking of the candidates retrieved across all granularities. Reducing this overhead is an engineering problem with principled directions on the software side. Distilled and compact cross-encoders, such as the MiniLM family [Wang et al. 2020], can lower reranking cost while preserving most of the relevance gains, and the greedy post-fusion deduplication described in Section 3 shortens the injected context. We regard these mitigations as future work and do not claim measured speedups here; for the high-stakes domains targeted by GRRF, the current trade-off favouring accuracy over latency is, in our view, a deliberate and defensible one.

6. Limitations

First, the evaluation compares the complete GRRF pipeline, which relies on a cross-encoder reranking stage, against single-pass bi-encoder baselines. This reflects a realistic comparison against typical out-of-the-box RAG configurations, but limits our ability to isolate the gains of multi-granularity fusion from the impact of the cross-encoder. Decoupling them through ablation is a necessary next step. We emphasize, however, that the reranker is applied uniformly within every granularity-specific list before fusion, so it is present in all internal branches of GRRF and cannot by itself explain the differences observed between granularities. Moreover, our central conclusion—that no single granularity is optimal across domains—is read directly from the baseline columns of Table 2

(512 wins in legal and news, 2048 wins in audit) and therefore already holds on the current data, independently of the reranked-baseline ablation, which we treat as work in progress.

Second, given the scarcity of public Portuguese benchmarks for structured domains, our evaluation relies on synthetically generated query-answer tuples. The protocol was controlled for traceability and validated by sampling, but synthetic queries inherit the linguistic patterns and biases of the generator model, and may not reproduce the ambiguity of real user interactions. Since GRRF is language-agnostic, evaluating it on established public benchmarks (e.g., BEIR, MS MARCO) is a natural next step that would enable direct comparison with the broader literature, which we leave as future work.

Finally, GRRF introduces computational and structural overheads: parallel retrievals plus a cross-encoder increase latency (a deliberate accuracy-over-speed trade-off), and retrieving multiple granularities from the same corpus can inject redundant, overlapping passages that consume the token budget and reduce context diversity—issues addressed respectively above.

7. Conclusion and Future Work

This paper showed empirically that no single chunking granularity is optimal across heterogeneous domains—a finding that holds directly among single-pass baselines and is therefore independent of reranking—and introduced Granularity Reciprocal Rank Fusion (GRRF), a retrieval strategy that operationalizes this finding by fusing multiple chunking granularities through supervised reranking and rank-based fusion instead of committing to one chunk size. Across legal, audit, and news domains, GRRF generalized well across heterogeneous document structures while improving retrieval quality over strong single-pass baselines.

To address computational overhead, we plan to explore lightweight and distilled reranking models to reduce inference time, making the solution viable for real-time, low-latency environments. To mitigate context redundancy, future investigations will explore alternative fusion techniques that explicitly promote context diversity alongside relevance, as well as dynamic chunking strategies that adapt to document characteristics to reduce reliance on manually defined segmentation scales.

Extending the evaluation to curated queries from domain experts is the next step towards assessing performance under practical constraints, particularly in sensitive domains where auditability matters.

Acknowledgments

This work was supported by Centro de Excelência em Inteligência Artificial. This work was also supported by the National Institute of Science and Technology (INCT) in Responsible Artificial Intelligence for Computational Linguistics and Information Treatment and Dissemination (TILD-IAR) grant number 408490/2024-1. The authors acknowledge the use of generative AI tools (Claude Opus 4.8 and Claude Opus 5, Anthropic) in the preparation of this manuscript. These tools were used for translation into English, grammar correction, and rephrasing for clarity, and to draft portions of the text from sources selected and supplied by the authors. No experimental results, analyses, or references were generated by these tools: all experiments were designed, executed, and interpreted

by the authors, and every cited reference was verified against the original source. All generated text was reviewed and edited by the authors, who take full responsibility for the final content.

References

Auer, C., Lysak, M., Nassar, A., Dolfi, M., Livathinos, N., Vagenas, P., Berrospi Ramis, C., Omenetti, M., Lindlbauer, F., Dinkla, K., Mishra, L., Kim, Y., Gupta, S., Teixeira de Lima, R., Weber, V., Morin, L., Meijer, I., Kuropiatnyk, V., and Staar, P. W. J. (2024). Docling technical report.

Bhat, S. R., Rudat, M., Spiekermann, J., and Flores-Herr, N. (2025). Rethinking chunk size for long-document retrieval: A multi-dataset analysis.

Chen, J., Xiao, S., Zhang, P., Luo, K., Lian, D., and Liu, Z. (2024). M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2318–2335, Bangkok, Thailand. Association for Computational Linguistics.

Chen, W., Bai, T., Su, J., Luan, J., Liu, W., and Shi, C. (2025). Kg-retriever: Efficient knowledge indexing for retrieval-augmented large language models.

Cormack, G. V., Clarke, C. L. A., and Büttcher, S. (2009). Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 758–759, New York, NY, USA. Association for Computing Machinery.

Dong, Z., Ni, J., Bikel, D., Alfonseca, E., Wang, Y., Qu, C., and Zitouni, I. (2022). Exploring dual encoder architectures for question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9419, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Es, S., James, J., Espinosa Anke, L., and Schockaert, S. (2024). 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*, pages 150–158, St. Julians, Malta. Association for Computational Linguistics.

Han, S. (2025). Finding the best chunking strategy for accurate ai responses. NVIDIA Technical Blog. Accessed: 2026-05-04.

Jégou, H., Douze, M., and Schmid, C. (2011). Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128.

Kane, A. (2024). pgvector: Open-source vector similarity search for Postgres. Software. Accessed: 2025-08-19.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

LangChain (2024). LangChain: A framework for developing applications powered by language models. Software. Accessed: 2025-08-19.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Lu, W., Chen, K., Qiao, R., and Sun, X. (2025). HiChunk: Evaluating and enhancing retrieval-augmented generation with hierarchical chunking.

Malkov, Y. A. and Yashunin, D. A. (2020). Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836.

Nguyen, H. T., Nguyen, T. D., and Nguyen, V. H. (2024). Enhancing retrieval augmented generation with hierarchical text segmentation chunking. In *International Symposium on Information and Communication Technology (SOICT 2024)*, Communications in Computer and Information Science, pages 209–220, Singapore. Springer Nature Singapore.

Rackauckas, Z. (2024). RAG-Fusion: A new take on retrieval-augmented generation. *International Journal on Natural Language Computing*, 13(1):37–47.

Santana, M. (2017). News of the Brazilian newspaper. Kaggle dataset. Accessed: 2025-08-19.

Sarthi, P., Abdullah, S., Tuli, A., Khanna, S., Goldie, A., and Manning, C. D. (2024). RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Schwaber-Cohen, R. and Patel, A. (2025). Chunking strategies for LLM applications. Pinecone Learn. Accessed: 2026-05-04.

Tao, W., Xing, X., Chen, Y., Huang, L., and Xu, X. (2025). TreeRAG: Unleashing the power of hierarchical storage for enhanced knowledge retrieval in long documents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 356–371, Vienna, Austria. Association for Computational Linguistics.

Toresan, B. R., Moreira, V. P., Paula, F. S. F., and Bencke, L. R. (2025). Advanced chunking techniques: a novel approach for semantic splitters. In *Anais do XL Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 455–465, Fortaleza, Brazil. SBC.

Tribunal de Contas dos Municípios do Estado de Goiás (2024). Portal de auditorias. Accessed: 2025-08-19.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., and Zhou, M. (2020). MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, Red Hook, NY, USA. Curran Associates Inc.

Zhong, Z., Liu, H., Cui, X., Zhang, X., and Qin, Z. (2025). Mix-of-granularity: Optimize the chunking granularity for retrieval-augmented generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5756–5774, Abu Dhabi, UAE. Association for Computational Linguistics.