Preliminary Ranking-Based Selection for Optimized Retriever Configuration in RAG Systems

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

[1] Informatics Institute, Federal University of Goiás (UFG),
Campus Samambaia - Alameda Palmeiras, s/n - Chácaras Califórnia,
Goiânia - GO, 74690-900, Brazil.

{tomazini, salvadorludovico}@discente.ufg.br

{savioteles, celso}@inf.ufg.br

Abstract. *Retrieval Augmented Generation (RAG) systems rely on efficient retrievers to fetch relevant documents, with their performance influenced by factors such as chunking methods and embedding models. These components determine how documents are segmented and semantically represented, directly impacting retrieval effectiveness. To enhance retriever performance, this study explores the construction of an optimizer capable of selecting the best configuration for document retrieval within a predefined solution space. The selection of technologies is a critical step, considering project constraints, query nature, and domain-specific requirements. A key challenge is filtering out unsuitable technologies while ensuring optimal performance.*

1. Introduction

The Retrieval-Augmented Generation (RAG) approach has emerged as a transformative solution in the field of natural language processing, significantly reshaping how artificial intelligence systems access and utilize knowledge [Lewis et al., 2020a, Karpukhin et al., 2020]. As organizations increasingly adopt these systems for critical applications in healthcare, legal services, education, and public information access [Lee et al., 2021, Wang and Zhao, 2023], the accuracy and reliability of information retrieval become essential elements that directly impact decision-making processes affecting millions of individuals.

However, the selection of optimal retrieval configurations remains a manual, trial-and-error process reliant on expert knowledge [Izacard and Grave, 2021, Thakur et al., 2022]. This not only creates barriers to adoption for non-expert organizations but also risks propagating misinformation, reinforcing biases, and failing to retrieve critical information in high-stakes domains [Bender et al., 2021, Sheng et al., 2019]. Therefore, developing systematic methods for optimizing retrieval components represents not only a technical advancement but a crucial step toward more accessible, fair, and responsible AI systems [Mitchell et al., 2019].

The effectiveness of a retriever component in RAG architectures heavily depends on two key elements: text segmentation strategies (chunking), which determine how documents are divided into smaller units, and embedding models, which encode semantic representations of these units [Luan et al., 2021, Yates et al., 2021]. Retrieval quality is directly linked to the appropriate choice of these two components, which must be selected considering factors such as computational constraints, budget limitations, query complexity, and domain-specific language. For instance, sentence-based strategies may fail in legal texts due to the frequent use of domain-specific abbreviations such as "art." or "sec.", which can mislead standard sentence segmentation algorithms [Chalkidis et al., 2020].

To address this challenge, we propose a retrieval optimizer designed to select the most appropriate configuration from a predefined set of chunking methods and embedding models. The process begins by filtering out options that are incompatible due to domain-specific or system-level constraints. Subsequently, a ranking-based evaluation strategy is applied to estimate the performance of each embedding model on the target dataset, thereby narrowing the search space to a promising subset of candidates. From this selection, the optimizer performs a detailed analysis to identify the configuration that maximizes retrieval quality while minimizing computational cost. This workflow aims to balance general performance with domain-specific effectiveness, ensuring robustness across diverse application contexts.

Beyond the standard execution pipeline, we also conduct a comprehensive experimental evaluation over all possible combinations of chunking and embedding, including those discarded in the preliminary stage. This additional grid search is not part of the core optimization pipeline, but serves to empirically validate the effectiveness of the ranking-based strategy by comparing automatically selected models with those that achieve absolute best performance.

This paper is organized as follows: Section 2 reviews the background and related work; Section 3 details the system architecture and optimization strategy; Section 4 describes the experimental setup and methodology; Section 5 presents the main results and supplementary analyses; Section 6 discusses the limitations of the proposed approach; and Section 7 concludes the paper and suggests directions for future work.

2. Background and Related Works

This section presents the foundational components of our approach and discusses related advancements, with a focus on text chunking strategies and embedding models in Retrieval-Augmented Generation (RAG) systems.

2.1. Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) combines information retrieval with generative models to enhance output accuracy and factual consistency [Lewis et al., 2020a]. The **Retriever** extracts relevant information from knowledge databases in response to queries. Its effectiveness depends critically on document chunking strategies and embedding quality [Zhong et al., 2025]. Retrieval approaches have evolved from token-level retrieval [Khandelwal et al., 2020] to entity-based [Nishikawa et al., 2022], chunk-based [Ram et al., 2023, Reimers and Gurevych, 2019], and graph-based methods [Kang et al., 2023].

2.2. Chunking Strategies

Text segmentation (*chunking*) significantly impacts RAG pipeline performance [Liu et al., 2025]. Basic approaches include *fixed-size* chunking (uniform blocks), *sliding window* with overlap [Safjan, 2023], *page-based* segmentation for PDFs, and *hierarchical* methods using structural delimiters. Advanced semantic strategies include *Parent Document Retrieval* [Antematter team, 2024], *Metadata Filtering*, and *Context-Enriched Chunking*.

2.3. Embedding Models

Dense vector retrieval maps texts into a continuous embedding space, positioning semantically similar documents and queries closer together for improved retrieval [Lewis et al., 2020b]. Current models like *E5* and *NV-Embed-v2* excel on MTEB benchmarks, while *KEPLER* enhances embeddings with knowledge graphs [Wang et al., 2021]. Although alternatives exist (BM25, ColBERT), our work focuses on dense embeddings.

3. System Architecture and Optimization Strategy Based on preliminary ranking

The primary objective of this work is to optimize document retrieval in Retrieval-Augmented Generation (RAG) systems by selecting the best combination of text segmentation strategy and embedding model. To this end, we propose a three-phase optimization architecture: (i) preliminary ranking of embedding models based on ranking metrics, (ii) document indexing using multiple text segmentation strategies and the top-n embedding models from phase (i), and (iii) exhaustive evaluation of the remaining combinations. The architecture code and datasets are publicly available [1].

3.1. Queries Generation

A crucial component of our evaluation framework is the creation of representative (query, reference document) pairs used to test retrieval quality. Synthetic queries are generated by applying LLMs to the original corpus with domain-specific prompts designed to emulate realistic user intent—for instance, prompts in the legal domain focus on case law, while those in auditing address compliance and internal controls. Each query is paired with a reference document selected for its semantic alignment and completeness in addressing the query, typically containing both the answer and contextual evidence. These pairs support the computation of standard retrieval metrics such as MRR and Recall@k under controlled yet realistic conditions. To enhance validity, we also incorporate a small set of real-world queries obtained through expert feedback and user studies, which provide qualitative insights into system behavior. All data (queries, references, and metadata) are stored in structured JSON files and publicly available in the project repository[1].

3.2. Preliminary Model Selection Phase

Prior to the complete execution of the optimization pipeline, an initial stage is carried out to filter out embedding models with unsatisfactory performance. This assessment is conducted using synthetic (query, reference) pairs, generated artificially, and applies classical Information Retrieval metrics to estimate each model's ranking capability. Each query represents a typical system input, while the reference document corresponds to the ideal retrieval outcome. The pairs are used to test the models' ability to correctly rank documents according to these metrics. Retrieval is performed, documents are ordered by similarity, and based on the position of the correct document, performance metrics are computed.

[1] `https://github.com/salvadorludovico/retriever-optmizer`

Specifically, two metrics are employed: Mean Reciprocal Rank (MRR) and Recall@k. The logic of this step is formalized in Algorithm 1. The implemented process is described below:

1. **Corpus Preparation:** Reference documents are extracted and deduplicated, forming the corpus that will serve as the knowledge base for retrieval.
2. **Similarity Calculation:** For each pair, similarity is computed between the query embedding and the document embeddings. The cosine similarity metric is adopted.
3. **Ranking and Evaluation:** Documents are sorted in descending order of similarity. The position of the reference document is recorded for the calculation of MRR and Recall@k.

Algorithm 1: Preliminary ranking-Based Evaluation for Embedding Selection

```
Input  : TOP_K, embedding model, data = [($query_1$, $reference_1$), ...]
Output: MRR, Recall
// Simulated data
corpus ← RemoveDuplicateReferences(data)
model ← LoadModel("embedding_model")
corpusEmbeddings ← EncodeCorpus(model, corpus)
results ← empty list
foreach (query, reference) ∈ data do
    queryEmbedding ← EncodeQuery(model, query)
    similarities ← ComputeCosineSimilarity(queryEmbedding,
      corpusEmbeddings)
    ranked ← SortDescending(similarities)
    refPosition ← FindPosition(ranked, reference)
    rank ← refPosition + 1
    Add (query, reference, rank, (rank $\le$ TOP_K), (1/rank)) to results
MRR ← AverageReciprocalRank(results)
Recall ← AverageRecallAtK(results)
```

This phase enables the selection of the most promising models based on their retrieval quality, thereby optimizing resources by avoiding the evaluation of all candidates during the final grid search.

3.3. Indexing Phase with Segmentation Strategies

Following the preliminary ranking of embedding models, the indexing phase begins. Each domain-specific dataset is segmented using multiple chunking methods, resulting in smaller text units. These chunks are then encoded with the embedding models selected in the previous phase, producing dense vector representations. The combination of a chunking method and an embedding model, referred to as a *configuration*, results in a unique collection of vectors, which is independently indexed in the vector store.

This design ensures that all relevant combinations of chunking and embedding are made available for retrieval during the evaluation phase, while restricting the scope exclusively to the embedding models identified as most promising in the previous step. The

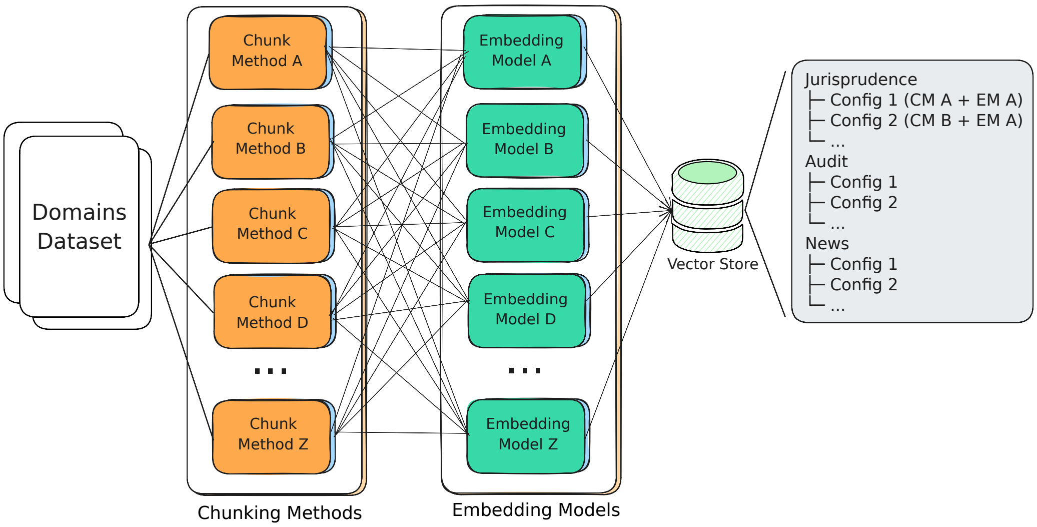

Figure 1. Indexing phase: document chunking and embedding generation.

overall workflow of this phase is depicted in **Figure 1**, which illustrates how each domain-specific dataset is segmented using multiple chunking strategies and subsequently encoded by each selected embedding model. Vector representations are stored in a domain-partitioned structure, with each collection corresponding to a specific configuration — defined by the combination of chunking strategy and embedding model. This schema supports flexible retrieval and evaluation, allowing independent access across different domains and configurations.

3.4. Evaluation and Recommendation Phase

The final stage of the architecture applies an exhaustive search (grid search) over the selected combinations of segmentation methods and embedding models derived from the preliminary ranking phase. The objective is to identify, within this reduced and promising subset, the best system configuration for each domain.

In this phase, synthetic queries are transformed into vectors using the same models employed during indexing. Each query vector is compared with the corresponding vector collections, and the five most similar documents are retrieved based on cosine similarity. The retrieved documents are then compared with the expected reference documents.

At the end of the evaluation, a decision function is applied to determine the most efficient configuration—that is, the combination of chunking method and embedding model that achieves the highest average performance within each domain. Configurations are grouped by arithmetic means of the obtained metrics, which are subsequently combined into a single value through a weighted average.

4. Experimental Materials and Methods

This section presents the datasets, experimental setup, and evaluation metrics used to validate the retrieval optimization strategies described in the previous section.

4.1. Datasets

We used three datasets from different domains, indexed in a vector store for retrieval tasks:

- **STJ Jurisprudence** — 8,000 documents containing decisions and interpretations from Brazil's Superior Court of Justice.
- **TCMGO Audit Reports** — 38 financial audit reports and rulings issued by Goiás' Municipal Court of Accounts.
- **Folha de S. Paulo Articles** — 8,000 news articles covering politics, economy, science, and society.

For each domain, we created a synthetic set of representative queries derived from document excerpts, aiming to simulate realistic user questions. In total, we generated 146 queries for Jurisprudence, 152 for Audit Reports, and 340 for News Articles. Each query was linked to a reference document to enable supervised evaluation. This association allowed us to accurately assess retrieval performance within each domain.

The **query** field represents the synthetic question, the **response_reference** specifies the ideal answer, and the **reference_passage** provides the supporting excerpt extracted from the original document.

These three fields are directly used in the computation of the evaluation metrics. The **Context Recall** metric assesses whether the relevant passages were successfully retrieved. The **Context Precision** metric measures the proportion of relevant passages among the top-K retrieved results. The **Faithfulness** metric verifies whether the generated answer is truly supported by the retrieved content.

4.2. Experimental Setup

The experiments were implemented in Python using LangChain and Hugging Face Transformers for embedding generation, with six sentence-transformer-based models evaluated. Models were selected based on multilingual support, instruction tuning, and recent benchmark performance. All models were run on a NVIDIA DGX station with two CUDA-enabled V100 GPUs (32 GB VRAM), allowing dynamic resource allocation and efficient parallel processing. Loading and preprocessing involved text extraction with `pdfplumber`, normalization, and structuring via `pandas`; `Pgvector` PostgreSQL extension was used for vector storage and search engine.

Document chunking was applied prior to vectorization, considering token limits and semantic coherence, supported by LangChain tools. Chunking methods were selected to balance granularity, structural awareness, and retrieval efficiency. Three embedding models—`BAAI/bge-m3`, `intfloat/multilingual-e5-large`, and `HIT-TMG/KaLM-embedding-multilingual-mini-instruct-v1.5`—were combined with segmentation strategies across three domains. For case law (STJ) and news articles (*Folha de S. Paulo*), five chunking methods were used: *hierarchical_sentence*, *sliding_window_1024*, *chunk_fixed_size_512*, *hierarchical_section*, and *hierarchical_paragraph*, resulting in 15 configurations per domain. For audit reports (TCM-GO), the same embeddings were tested including an additional method, *per_page_chunk*, resulting in a total of 18 combinations.

4.3. Evaluation metrics

To aggregate the metrics, we applied a weighted average, with weights defined through a principled qualitative analysis conducted by the authors. This analysis was grounded in the practical role and relative importance of each metric in RAG-based systems. Context Recall (0.35) received the highest weight due to its strong influence on answer completeness, followed by Faithfulness (0.30), which ensures answer reliability. Context Precision was split into two variants: 0.25 for the reference-based version, considered more stable, and 0.10 for the no-reference variant. These choices reflect an informed understanding of the trade-offs involved in retrieval-based evaluation and aim to balance completeness, precision, and factual grounding, [Es et al., 2023].

5. Results and Discussion

This section presents and analyzes the results obtained through the proposed optimization pipeline, with particular emphasis in the preliminary embedding model ranking stage. The discussion is structured by domain — audit, news, and legal — and evaluates whether the initial ranking metrics (Mean Reciprocal Rank and Recall@5) effectively anticipated the best-performing configurations in a full grid search.

To validate this strategy, we compare the ordering induced by the preliminary ranking with the complete performance results obtained for each model-domain pair. These results are summarized in Table 1, which serves as a consolidated reference throughout this section. This table reports the MRR and Recall@5 values for the three top-ranked models evaluated via full grid search, across the three domains. It allows us to assess whether the embedding models prioritized in the early phase remain superior when combined with optimal chunking strategies.

The subsequent subsections examine these findings in detail, highlighting the alignment (or deviations) between early ranking metrics and actual downstream performance, and assessing the consistency and generalizability of the optimizer across domains.

5.1. News Domain Performance

In the news domain, experiments were conducted with the three embedding models previously selected through preliminary ranking: **BAAI/bge-m3**, **KaLM**, and **multilingual-e5-large**. The objective was to assess whether the top-ranked models based on initial metrics would also maintain superior performance in a full grid search with different text segmentation strategies.

The complete combination results are shown in **Table 2**. The best configuration was obtained using the **BAAI/bge-m3** model with the *sliding-window-1024* segmentation, achieving a *Weighted Average* (WA) of **86.77%** and a *Context Recall* (CR) of **92.51%**. Other configurations with the same model, such as *hierarchical-section*, also yielded strong results (WA = 83.58%). The **multilingual-e5-large** model, while showing relatively good performance with *sliding-window-1024* (WA = 84.39%), remained behind BGE-M3. The **KaLM** model achieved WA = 83.84% with the same segmentation but showed higher sensitivity to changes in granularity, as evidenced by significant performance drops with more fragmented strategies such as *hierarchical-paragraph* and *hierarchical-sentence* (WA below 60%).

The correspondence between grid search results and the aggregated preliminary ranking values per model, as reported in **Table 1**, reinforces the validity of the selection strategy. The **BAAI/bge-m3** model, which achieved the highest preliminary metrics in the news domain (MRR = 88.74; Recall@5 = 94.12), also ranked first in the full evaluations. The relative ordering among models was preserved, with **KaLM** and **multilingual-e5-large** performing slightly lower, though still competitive in specific configurations.

Table 1. Performance of embedding models by domain

Domain	Model	MRR (%)	Recall@5 (%)
Jurisprudence	`BAAI/bge-m3`	**91.51**	**97.95**
	`multilingual-e5-large`	89.37	95.21
	`KaLM`	86.18	94.52
Audit	`BAAI/bge-m3`	**50.67**	**63.82**
	`KaLM`	45.99	61.84
	`multilingual-e5-large`	47.48	60.53
News	`BAAI/bge-m3`	**88.74**	**94.12**
	`KaLM`	77.70	92.94
	`multilingual-e5-large`	75.89	85.88

5.2. Legal Domain Performance

The legal domain comprises documents such as court rulings and legal opinions, characterized by highly formal structure, high informational density, and strong semantic continuity. In this setting, the three embedding models selected via preliminary ranking — **BAAI/bge-m3**, **KaLM**, and **multilingual-e5-large** — were evaluated in a full grid search across various segmentation methods.

According to **Table 4**, the best observed configuration was **BAAI/bge-m3 + chunk-fixed-size-512**, which yielded a WA of **91.49%** and CR of **96.99%**. Other segmentations applied to the same model, such as *hierarchical-paragraph* (WA = 90.99%) and *sliding-window-1024* (WA = 90.03%), also performed well, demonstrating the robustness of BGE-M3 across different granularities. The **multilingual-e5-large** and **KaLM** models also showed competitive performance in structured configurations. For example, **multilingual-e5-large** achieved WA = 90.02% with *hierarchical-paragraph*, while **KaLM** reached WA = 87.27% using the same strategy. However, these models exhibited greater variability depending on the segmentation, with sharper declines in performance using less cohesive methods such as *chunk-fixed-size-512* or *hierarchical-sentence*.

Once again, the alignment between grid search results and the preliminary ranking metrics per model, shown in **Table 1**, validates the preliminary ranking step. The **BAAI/bge-m3** model maintained its lead, achieving the highest MRR (91.51) and Recall@5 (97.95) in the legal domain. The other models followed the same relative performance order, with **multilingual-e5-large** in second and **KaLM** in third.

5.3. Audit Domain Performance

In the audit domain, a full grid search was conducted using the three embedding models selected in the preliminary phase: **BAAI/bge-m3**, **KaLM**, and **multilingual-e5-large**. The objective was to validate the effectiveness of the preliminary stage as a filtering mechanism by evaluating whether the models selected using *Mean Reciprocal Rank* (MRR) and *Recall@5* would continue to perform best in exhaustive evaluation across multiple segmentations.

Table 2. The following table has the results of each combination for the Folha de S. Paulo News Articles dataset. Weighted Average (WA), Faithfulness (F), Context Recall (CR), Context Precision (CP). Bold values denote the highest score per column.

Embedding Model	Chunk Method	WA (%)	F (%)	CR (%)	CP (Ref) (%)	CP (No Ref) (%)
BAAI/bge-m3	sliding-window-1024	**86.77**	**86.03**	**92.51**	**81.64**	**81.73**
multilinguale5large	sliding-window-1024	84.39	85.27	89.89	78.04	78.36
KaLM	sliding-window-1024	83.84	83.26	89.39	78.67	79.11
BAAI/bge-m3	hierarchical-section	83.58	80.14	91.62	78.56	78.3
multilinguale5large	hierarchical-section	75.72	73.17	81.33	72.32	72.25
BAAI/bge-m3	chunk-fixed-size-512	68.89	61.93	74.53	69.25	69.12
BAAI/bge-m3	hierarchical-paragraph	66.12	58.27	70.74	68.12	68.53
KaLM	hierarchical-section	65.49	61.87	71.61	62.04	63.54
KaLM	chunk-fixed-size-512	59.34	52.55	64.27	60.5	59.52
KaLM	hierarchical-paragraph	57.84	49.8	62.3	60.37	60.04
multilinguale5large	chunk-fixed-size-512	56.91	50.8	62.95	55.95	56.53
multilinguale5large	hierarchical-paragraph	56.78	50.09	57.83	61.06	62.51
BAAI/bge-m3	hierarchical-sentence	48.84	36.99	47.23	60.93	59.82
multilinguale5large	hierarchical-sentence	44.5	35.01	42.63	54.3	54.98
KaLM	hierarchical-sentence	41.34	29.98	39.79	52.47	52.99

Detailed results are presented in **Table 5**. The top-performing configuration was **BAAI/bge-m3 + sliding-window-1024**, which achieved WA = **84.64%** and CR = **92.3%**. Similar configurations such as *per-page-chunk* and *hierarchical-section* also showed strong performance with the same model, confirming its robustness across granularities. In contrast, **KaLM** and **multilingual-e5-large** consistently underperformed. Even their best combinations — **KaLM + per-page-chunk** (WA = 81.32%) and **multilingual-e5-large + sliding-window-1024** (WA = 79.1%) — fell short of BGE-M3's top configurations. Furthermore, all models exhibited substantial performance drops with the *hierarchical-sentence* strategy, indicating that inadequate granularity negatively impacts retrieval in long and technically dense audit texts.

The relationship between the grid search results and the preliminary ranking values per model, as shown in **Table 1**, supports the utility of the preliminary ranking phase. The **BAAI/bge-m3** model, which led in the initial metrics (MRR = 50.67; Recall@5 = 63.82), also achieved the best grid search outcomes. The relative performance order of **KaLM** and **multilingual-e5-large** was maintained. These results demonstrate that the preliminary ranking stage was effective in narrowing the search space without excluding the best-performing model, enabling a more efficient and targeted optimization process. In the audit domain, marked by long and interdependent documents, the strategy successfully preserved high-quality configurations while empirically discarding suboptimal ones.

To examine the generalizability of the preliminary ranking strategy, an additional experiment was conducted in the audit domain using a distinct set of embedding models: **nomic-embed-text-v1**, **mxbai-embed-large-v1**, and **flan-t5-large**. These models span different architectures, parameter scales, and training objectives. The preliminary evaluation was based on MRR and Recall@5, with results shown in **Table 3**. The **nomic-**

embed-text-v1 model achieved the highest values (MRR = 0.3992; Recall@5 = 52.63%), followed by **mxbai-embed-large-v1** (MRR = 0.3122; Recall@5 = 37.50%) and **flan-t5-large** (MRR = 0.2124; Recall@5 = 28.95%). To estimate downstream performance without applying a full grid search, we used the *sliding-window-1024* configuration as the baseline in the optimizer.

Table 3. Performance of embedding models using MRR, Recall@5, and weighted average (without optimizer).

Model	MRR	Recall@5	Weighted Avg.
nomic-embed-text-v1	**0.3992**	**52.63%**	~76.7%
mxbai-embed-large-v1	0.3122	37.50%	~69.3%
flan-t5-large	0.2124	28.95%	~53.6%

This segmentation was previously identified as effective in the audit domain (see **Table 5**), enabling consistent estimation of each model's WA. The resulting values were **76.7%** for **nomic-embed-text-v1**, **69.3%** for **mxbai-embed-large-v1**, and **53.6%** for **flan-t5-large**. This strategy enables efficient screening of promising models without the need for exhaustive evaluation, maintaining strong alignment with performance estimated using a reference configuration.

5.4. Analysis of Embedding Model Performance: The Case of BGE-M3

The BAAI/bge-m3 model demonstrated superior performance across all evaluated domains. BGE-M3 is a hybrid embedding model designed to support dense, sparse, and multi-vector retrieval modes within a unified architecture Liu et al. [2024]. This design allows the model to jointly capture semantic similarity, lexical matching, and multi-faceted contextual signals, which is particularly beneficial in domains with dense informational content, such as legal and audit documents. Unlike traditional single-vector encoders, BGE-M3 produces multiple representations per input, improving retrieval robustness across varied query types. A distinguishing feature of BGE-M3 is its support for long input sequences of up to 8192 tokens, facilitated by efficient multi-CLS pooling mechanisms. This capacity enables effective use of large-span chunking strategies, such as sliding-window with 1024-token stride, without significant loss in representation quality. In our experiments, this characteristic aligned with its strong performance across segmentation methods, particularly in the news and audit domains.

The training process of BGE-M3 employs self-knowledge distillation, wherein different retrieval heads within the model supervise each other to produce internally consistent embeddings Liu et al. [2024]. This strategy reduces overfitting to any single retrieval mode and improves generalization across domains. Furthermore, the model was trained on large-scale, multi-domain corpora using contrastive objectives, enhancing its domain transferability. These design decisions are reflected in public benchmark results. On the MTEB leaderboard Muennighoff et al. [2023], BGE-M3 consistently outperforms models such as multilingual-e5-large and KaLM in retrieval tasks, including in multilingual and legal settings. Independent comparisons further confirm its competitiveness against proprietary models such as OpenAI's text-embedding-ada-002 Gupta [2024].

In summary, BGE-M3's multi-representation architecture, long-context handling, and distillation-based training collectively explain its superior performance in both preliminary ranking metrics and downstream evaluation across diverse document domains.

6. Limitations of the preliminary ranking Strategy

Although the results presented confirm the effectiveness of the preliminary ranking strategy in reducing the search space and preserving high-performing configurations, this approach presents important limitations that must be considered in practical applications.

The primary limitation lies in the **direct dependence on the quality and representativeness of the synthetic query-reference set** used in the initial evaluation phase. Since the selection of embedding models is guided by ranking metrics computed from this set, any imbalance, thematic bias, or limited semantic coverage may compromise the accuracy of the pre-selection. If the queries do not adequately reflect the complexity, style, or informational scope of real-world documents in the target domain, the selected model may not be the most appropriate for actual usage scenarios, even if it performs well in preliminary metrics. Moreover, since synthetic data are derived from the same document corpus, there is a risk of local overfitting — that is, a model may rank extracted pairs effectively but fail to generalize to real or future system inputs.

Table 4. The following table has the results of each combination for the STJ Jurisprudence dataset. Weighted Average (WA), Faithfulness (F), Context Recall (CR), Context Precision (CP). Bold values denote the highest score per column.

Embedding Model	Chunk Method	WA (%)	F (%)	CR (%)	CP (Ref) (%)	CP (No Ref) (%)
BAAI/bge-m3	chunk-fixed-size-512	**91.49**	83.98	**96.99**	92.51	92.25
BAAI/bge-m3	hierarchical-paragraph	90.99	**86.21**	92.94	**93.04**	**93.39**
BAAI/bge-m3	sliding-window-1024	90.03	86.12	94.79	88.63	88.56
multilinguale5large	hierarchical-paragraph	90.02	84.15	94.56	90.46	90.59
KaLM	hierarchical-paragraph	87.27	79.09	92.48	89.19	88.8
multilinguale5large	sliding-window-1024	87.01	82.99	94.91	82.26	83.29
BAAI/bge-m3	hierarchical-sentence	86.75	77.66	88.31	92.93	93.14
multilinguale5large	hierarchical-sentence	86.16	76.54	89.81	90.5	91.38
multilinguale5large	chunk-fixed-size-512	85.98	77.78	91.78	87.05	87.61
KaLM	sliding-window-1024	85.82	83.8	93.52	79.89	79.77
BAAI/bge-m3	hierarchical-section	85.36	79.82	92.48	82.98	83.07
multilinguale5large	hierarchical-section	82.05	77.88	91.32	76.32	76.41
KaLM	hierarchical-sentence	81.48	70.57	84.61	87.83	87.36
KaLM	chunk-fixed-size-512	80.62	72.3	86.92	81.45	81.43
KaLM	hierarchical-section	60.83	54.16	70.14	57.2	57.34

Therefore, the **effectiveness of the preliminary ranking critically depends on the careful construction of the query-reference dataset**, which must be sufficiently diverse, contextually relevant, and aligned with the application domain. Automated strategies for query generation, enrichment with real usage examples, and iterative evaluation of semantic coverage are promising directions to mitigate this limitation and enhance the robustness of the method across different contexts.

7. Conclusion

This work proposed a modular framework for optimizing retrievers in Retrieval-Augmented Generation (RAG) systems, based on the preliminary ranking of embedding models using classical ranking metrics. The central objective was to reduce the search space during exhaustive configuration exploration while preserving the quality of the resulting solutions. The architecture was structured into three stages — preliminary ranking, segmented indexing, and exhaustive evaluation — enabling the application of progressive filters to efficiently identify the best combination of embedding model and text segmentation strategy.

The pre-selection stage, guided by metrics such as Mean Reciprocal Rank (MRR) and Recall@k, proved to be an effective tool for anticipating downstream retrieval performance. Experiments conducted across three distinct domains — audit, news, and legal — empirically validated this approach, showing that the top-ranked models in preliminary evaluations maintained superior performance in subsequent stages, even under varying granularities and heterogeneous textual structures. The main contributions of this study are: (i) the formalization of an optimization pipeline for retrievers based on modular, empirically-driven stages; (ii) the introduction of a lightweight preliminary ranking mechanism that significantly reduces the search space; and (iii) the experimental validation of the approach in diverse scenarios, demonstrating its generalizability and practical applicability.

Table 5. The following table have the results of each combination for the TCMGO Audit Reports dataset. Weighted Average (WA), Faithfulness (F), Context Recall (CR), Context Precision (CP). Bold values denote the highest score per column.

Embedding Model	Chunk Method	WA (%)	F (%)	CR (%)	CP (Ref) (%)	CP (No Ref) (%)
BAAI/bge-m3	sliding-window-1024	**84.64**	**79.42**	**92.3**	81.47	81.37
BAAI/bge-m3	per-page-chunk	83.19	75.53	90.11	**82.66**	**83.33**
BAAI/bge-m3	hierarchical-section	81.53	74.03	87.09	82.33	82.58
KaLM	per-page-chunk	81.32	72.66	89.87	79.83	81.13
KaLM	sliding-window-1024	81.29	73.25	88.68	81.08	80.09
multilinguale5large	sliding-window-1024	79.1	70.9	90.57	74.71	74.5
KaLM	hierarchical-section	74.96	65.06	82.46	75.84	76.25
BAAI/bge-m3	chunk-fixed-size-512	72.33	57.09	81.03	76.78	76.5
multilinguale5large	per-page-chunk	68.81	58.95	81.2	64.64	65.43
BAAI/bge-m3	hierarchical-sentence	67.56	52.67	75.72	72.23	71.94
BAAI/bge-m3	hierarchical-paragraph	67.4	48.26	76.9	74.38	74.13
multilinguale5large	chunk-fixed-size-512	67.14	51.32	78.38	69.22	70.05
KaLM	chunk-fixed-size-512	66.08	46.36	76.15	73.17	72.27
KaLM	hierarchical-paragraph	65.04	46.01	77.52	68.56	69.63
multilinguale5large	hierarchical-section	64.9	55.27	77.25	60.64	61.25
multilinguale5large	hierarchical-paragraph	62.66	42.38	75.67	67.22	66.51
KaLM	hierarchical-sentence	57.46	35.9	67.93	65.87	64.52
multilinguale5large	hierarchical-sentence	55.31	35.67	61.98	65.52	65.36

As future work, we propose to explore extensions of the optimizer through hybrid retrieval approaches (e.g., sparse or supervised models) and, in later stages, develop complementary mechanisms for answer generation, integrating a Generation Optimizer into the full RAG system pipeline.

Acknowledgments

We acknowledge the support provided by the Center of Excellence in Artificial Intelligence (CEIA). We express gratitude to the Court of Accounts of State of Goiás Municipalities (TCM-GO) and the Court of Accounts of the State of Goiás (TCE-GO), whose generous contributions were fundamental to the development of this work. 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.

References

Antematter team. Optimizing retrieval-augmented generation with advanced chunking techniques: A comparative study, 2024. Accessed: 2025-03-31.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of FAccT*, 2021.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. LEGAL-BERT: The muppets straight out of law school. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.261. URL `https://aclanthology.org/2020.findings-emnlp.261/`.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. Ragas: Automated evaluation of retrieval augmented generation, 2023. URL `https://doi.org/10.48550/arXiv.2309.15217`.

Naman Gupta. Bge-m3 vs openai embeddings: A comparative study. `https://naman1011.medium.com/bge-m3-model-vs-openai-embeddings-e6d6cda27d0c`, 2024.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of EACL*, 2021.

Jungwoo Kang, Jinhyuk Lee, and Jaewoo Kang. Knowledge graph-augmented language models for knowledge-grounded dialogue generation. *arXiv preprint arXiv:2305.18846*, 2023.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of EMNLP*, 2020.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*, 2020.

Joon Lee, Hyoungho Yoon, and Hyeoun-Ae Park. Explainable ai in healthcare: From black box to interpretable models. *Healthcare Informatics Research*, 27(1):1–9, 2021.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Angela Fan, Vishrav Chaudhary, Tim Rocktäschel, and Sebastian Riedel. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, 2020a.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2020b. URL `https://doi.org/10.48550/arXiv.2005.11401`.

Xiao Liu, Zihan Zhou, Tianyu Zhou, Maosong Sun, and Tianyu Wang. Bge-m3: A multi-function embedding model for dense, sparse and multi-vector retrieval. *arXiv preprint arXiv:2402.03216*, 2024. URL `https://arxiv.org/abs/2402.03216`.

Zuhong Liu, Charles-Elie Simon, and Fabien Caspani. Passage segmentation of documents for extractive question answering, 2025. URL `https://doi.org/10.48550/arXiv.2501.09940`.

Yi Luan, Kaitao Tang, Mandar Joshi Gupta, and Luke Zettlemoyer. Sparse retrieval for question answering. In *Proceedings of ACL*, 2021.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2019.

Niklas Muennighoff, Nizar Tazi, et al. Mteb: Massive text embedding benchmark. `https://huggingface.co/spaces/mteb/leaderboard`, 2023.

Taichi Nishikawa, Soichiro Hidaka, Sho Yokoi, and Hideki Nakayama. Towards entity-enhanced RAG: Augmenting retrieval augmented generation with entity annotation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *Transactions on Machine Learning Research*, 2023.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

Krystian Safjan. From fixed-size to nlp chunking - a deep dive into text chunking techniques, 2023. Accessed: 2025-03-31.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. The woman worked as a babysitter: On biases in language generation. In *Proceedings of EMNLP*, 2019.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2022.

Rui Wang and Lili Zhao. Ai in education and policy-making: A review of recent advances. *Educational Technology Research and Development*, 71(2):135–152, 2023.

Xiang Wang, Xiangyu Dong, Fuzheng Zhang, Liwei Wang, and Xing Xie. Kepler: A unified model for knowledge embedding and pre-trained language representation, 2021. URL `https://arxiv.org/abs/1911.06136`. BLOG / Survey style reference.

Andrew Yates, Sebastian Hofstätter, and Guido Zuccon. Pretrained transformers for text ranking: Bert and beyond. *arXiv preprint arXiv:2104.08663*, 2021.

Zijie Zhong, Hanwen Liu, Xiaoya Cui, Xiaofan Zhang, and Zengchang Qin. Mix-of-granularity: Optimize the chunking granularity for retrieval-augmented generation, 2025. URL `https://doi.org/10.48550/arXiv.2406.00456`.