Beyond Vector Search: Comparing Classical RAG with Hybrid GraphRAG for Climate Science Q&A

Daniel Naiff[1], **Ronnie Alves**[2], **Gustavo Pinto**[1]

[1] Universidade Federal do Pará (UFPA)

[2]Instituto de Tecnologia Vale (ITV)

danielnaiff344@gmail.com, ronnie.alves@itv.org, gpinto@ufpa.br

Abstract. *Traditional Retrieval-Augmented Generation (RAG) systems treat documents in isolation, failing to capture hierarchical relationships between concepts in complex scientific corpora. This limitation compromises answer quality in specialized domains such as climatology, where conceptual dependencies frequently traverse multiple articles. We propose a hybrid architecture that integrates vector search with GraphRAG, Leiden community detection, and cross-encoder re-ranking, achieving gains of 160% in contextual relevance and 177% in contextual recall compared to classical RAG. These results demonstrate that unifying local and global retrieval significantly outperforms text-span isolation, paving the way for more effective question-answering systems over dispersed scientific literature.*
Keywords: *RAG, GraphRAG, Information Retrieval.*

Resumo. *Sistemas de recuperação aumentada por geração (RAG) tradicionais tratam documentos de forma isolada, falhando em capturar relações hierárquicas entre conceitos em corpora científicos complexos. Essa limitação compromete a qualidade das respostas em domínios especializados como climatologia, onde dependências conceituais frequentes atravessam múltiplos artigos. Propusemos uma arquitetura híbrida que integra busca vetorial com GraphRAG, detecção de comunidades Leiden e re-ranking via cross-encoder, alcançando ganhos de 160% em relevância contextual e 177% em recall contextual em relação ao RAG clássico. Esses resultados demonstram que a unificação de recuperação local e global supera significativamente o isolamento de trechos textuais, abrindo caminhos para sistemas de Q&A mais eficazes em acervos científicos dispersos.*
Palavras-chave: *RAG, GraphRAG, Recuperação de Informação.*

1. Introdução

A Amazônia desempenha um papel fundamental no equilíbrio climático global, atuando como sumidouro de carbono e reguladora dos regimes de precipitação na América do Sul. Esse destaque tem impulsionado um crescimento substancial na produção de artigos científicos sobre o bioma. Embora positivo, esse volume de publicações cria um desafio para sistemas de perguntas e respostas: extrair conhecimento relevante a partir de documentos dispersos.

Os sistemas de Geração Aumentada por Recuperação (*Retrieval Augmented Generation*, *RAG*) surgiram como solução promissora ao combinar *Large Language Models*

(LLMs) com busca vetorial. No entanto, a abordagem vetorial tradicional trata os documentos de forma isolada, sem capturar relações hierárquicas entre conceitos. Trabalhos recentes propõem evoluções dessa arquitetura, como os baseados em grafos de conhecimento (*GraphRAG*), que demonstram potencial para superar tais limitações ao extrair entidades, relações e comunidades de conhecimento.

Este artigo compara o *RAG* tradicional com o *GraphRAG* [Edge et al. 2024], uma de suas ramificações mais especializadas. As abordagens foram aplicadas em um sistema de perguntas e respostas cuja base de dados consiste em artigos científicos produzidos por um grupo de pesquisa do Instituto de Tecnologia Vale (ITV) (mais detalhes na Seção 3). Para os experimentos, foi curado um dataset com 22 perguntas e suas respostas esperadas pela equipe de pesquisadores usuárias do sistema. Esse dataset foi avaliado com 7 LLMs de diferentes tamanhos e arquiteturas, utilizando métricas baseadas em *LLM-as-a-Judge* como *correctness*, *answer relevance* e *faithfulness*. Os resultados indicam que a abordagem híbrida supera significativamente o *RAG* vetorial tradicional, com ganhos de 160% em relevância contextual e 177% em *recall contextual*.

2. Arquiteturas RAG e GraphRag

A arquitetura *RAG* foi proposta por [Lewis et al. 2020] unindo modelos de linguagem e recuperação de informação. No *RAG* clássico, os documentos são indexados em *embeddings* vetoriais; na consulta, os *chunks* mais semelhantes semanticamente à pergunta compõem o contexto do gerador. Embora eficaz, essa abordagem falha em modelar relações explícitas entre conceitos ou documentos isolados, motivando extensões estruturais. O *GraphRAG*, por outro lado, proposto por [Edge et al. 2024], supera essa limitação ao integrar grafos de conhecimento. A abordagem extrai entidades e relações textuais via LLM, aplicando o algoritmo Leiden para detectar comunidades hierárquicas e gerar sumarizações multi-nível. Isso permite ao sistema explorar conexões globais e distribuídas no *corpus*. O ganho dessa representatividade estrutural é respaldado por [Sequeda et al. 2024], que demonstram que consultas *zero-shot* em dados brutos atingem apenas 16% de acurácia, ao passo que a introdução de uma camada semântica baseada em grafos de conhecimento eleva essa métrica para 54%.

Esses achados demonstram que o raciocínio das LLMs é limitado, sem conexões explícitas. Na análise do acervo científico do ITV, a necessidade de correlacionar temas transversais replica esse desafio. Portanto, a escolha do *GraphRAG* tende-se a justificar por ir além da similaridade geométrica de palavras-chave, utilizando o grafo como um mapa conceitual de alta precisão para guiar a geração de respostas.

3. Arquiteturas propostas

Esta seção descreve as duas arquiteturas desenvolvidas para o chatbot inteligente especializado na consulta e síntese de artigos científicos da instituição parceira. A primeira versão foi implementada com base na abordagem clássica de RAG. A segunda versão aprimora a arquitetura inicial ao incorporar técnicas de *GraphRAG*, visando enriquecer o contexto recuperado, além da injeção do título do artigo em cada *chunk* gerado durante o processo de indexação vetorial.

A escolha pela abordagem baseada em grafos justifica-se pela necessidade de ir além da similaridade geométrica isolada de palavras-chave da busca vetorial tradicio-

nal, mapeando explicitamente o conhecimento correlacionado entre múltiplos documentos científicos sobre a climatologia da Amazônia. Neste cenário estruturado, os nós (ou vértices) do grafo representam as entidades textuais extraídas dos documentos (tais como conceitos climáticos, localizações geográficas, metodologias, autores e instituições), enquanto as arestas (ou vínculos) mapeiam as relações semânticas e conexões explícitas existentes entre essas entidades (como, por exemplo, o impacto de uma variável meteorológica sobre uma região específica ou a coautoria de um estudo). Essa modelagem estruturada permite que o algoritmo de detecção de comunidades organize o conhecimento de forma macroscópica e hierárquica, mitigando o isolamento de passagens textuais.

O sistema proposto integra: (i) um pipeline de ingestão que converte PDFs, extrai metadados, traduz conteúdo para português, gera *chunks* e *embeddings*, constrói um grafo de conhecimento com detecção hierárquica de comunidades e sumarização *bottom-up*; e (ii) um mecanismo de consulta que unifica a busca vetorial e os resumos de comunidades em um processo de *reranking* com *cross-encoder* multilíngue, garantindo pesos iguais às duas fontes de informação. Por diretriz de design da arquitetura e de projeto, todas as soluções tecnológicas e modelos empregados no ecossistema do sistema proposto foram baseados em ferramentas e tecnologias de código aberto (*open-source*), garantindo a reprodutibilidade, autonomia e controle sobre a privacidade e o processamento local dos dados científicos.

3.1. Arquitetura Geral, v1

A Figura 1 ilustra o pipeline de ingestão da primeira versão do sistema. Este pipeline é responsável pelo processamento dos documentos PDF e pela construção da base de conhecimento vetorial.

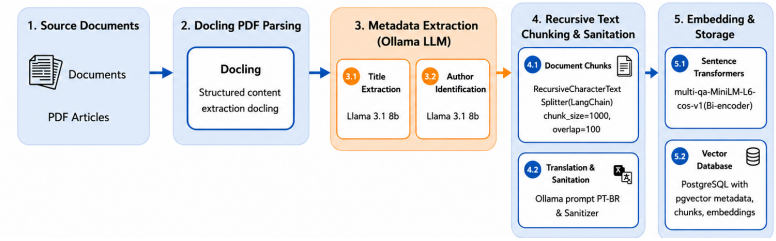


Figura 1. Pipeline de ingestão da primeira versão do chatbot.

O fluxo inicia-se com a etapa 1 (*Source Documents*), contendo os artigos em formato PDF. Em seguida, na etapa 2 (*Docling PDF Parsing*), ocorre a extração estruturada do conteúdo dos arquivos PDF por meio da biblioteca `Docling`[1]. Na etapa 3 (*Metadata Extraction (Ollama LLM)*), a primeira página de cada artigo é processada para a extração automática do título do artigo e informação dos autores além de outros metadados. Para essa extração, são utilizados prompts específicos enviados ao modelo `llama 3.1 8b` via Ollama. Sendo o mapeamento do título orientado pela subetapa 3.1 (*Title Extraction*).

Posteriormente, na etapa 4 (*Recursive Text Chunking & Sanitation*), o texto completo do documento é segmentado na subetapa 4.1 (*Document Chunks*) por meio da técnica de *chunking* recursivo[2]. O *chunking* recursivo funciona de forma hierárquica, utilizando uma lista de separadores padrão — iniciando por quebras de parágrafo (“`\n\n`”),

[1] `https://github.com/DS4SD/docling`

[2] Implementado pela classe `RecursiveCharacterTextSplitter`, do framework `LangChain`.

seguidas por quebras de linha simples ("\n"), espaços em branco (" ") e, por fim, caracteres vazios ("") —, dividindo o texto de maneira recursiva sempre que um bloco excede o limite máximo configurado, o que garante a preservação da integridade estrutural de frases e parágrafos completos. Após alguns experimentos, foram definidos os valores de 1.000 caracteres e sobreposição de 100 caracteres. Isso significa que cada fragmento gerado conterá no máximo 1.000 caracteres e, para evitar a perda de contexto nas bordas do corte, os últimos 100 caracteres de um bloco são replicados no início do bloco seguinte, mantendo a continuidade semântica entre eles. Após o processo de segmentação, na subetapa 4.2 (*Translation & Sanitation*), cada *chunk* gerado passa por um processo de sanitização e tradução automática para o português. Isso é feito pois o alinhamento idiomático entre a base de dados e as consultas do usuário otimiza a recuperação e garante que os *embeddings* capturem com maior precisão as relações semânticas na língua nativa do sistema final, reduzindo o custo e o tempo de processamento em tempo de execução.

Após a tradução, na etapa 5 (*Embedding & Storage*), os *chunks* são codificados em vetores de *embeddings* na subetapa 5.1 (*Sentence Transformers*[3]) utilizando o modelo *bi-encoder* `multi-qa-MiniLM-L6-cos-v1` da biblioteca *Sentence Transformers*. Por fim, na subetapa 5.2 (*Vector Database*), os metadados (título e autores), as páginas do artigo segmentadas em *chunks*, e seus respectivos embeddings são armazenados no banco de dados *PostgreSQL*, utilizando a extensão *pgvector*[4] para armazenamento vetorial.

3.1.1. Recuperação de informação

A Figura 2 apresenta o pipeline de recuperação de informações e geração de respostas da primeira versão do *chatbot*.

Na etapa 1 (*Input & Classification*), ao receber a pergunta do usuário, o sistema inicia o processo de recuperação contextualizada. Primeiramente, na etapa 2 (*Document Retrieval*), a pergunta do usuário é codificada em um vetor de *embedding* na subetapa 2.1 (*Bi-Encoder*) utilizando o mesmo *bi-encoder* (`multi-qa-MiniLM-L6-cos-v1`). Em seguida, na subetapa 2.2 (*Vector Search*), é realizada uma busca de similaridade no banco, utilizando distância euclidiana como método de comparação de distâncias. Recupera-se os 10 *chunks* mais semelhantes.

Na etapa 3 (*Re-ranking*), os chunks recuperados passam por uma segunda etapa de *reranking* na subetapa 3.1 (*Cross-Encoder*) utilizando o modelo *Cross-Encoder* `cross-encoder/stsb-distilroberta-base`. Esta etapa seleciona somente so dois *chunks* mais relevantes com base na similaridade fina entre a pergunta do usuário e cada *chunk* candidato.

Com o contexto recuperado, na etapa 4 (Answer Generation), o sistema preenche dinamicamente um *template* de *prompt* predefinido, injetando as instruções estáticas de comportamento, os dois *chunks* reranqueados na subetapa 4.2 (Context) e a pergunta do usuário. Esse *prompt* devidamente estruturado é então enviado na subetapa 4.1 (*LLM*

[3]Biblioteca Python para geração de embeddings de sentenças baseada em modelos Transformer. Disponível em: `https://www.sbert.net/`. Acesso em: 22 jun. 2026.

[4]Extensão *open-source* para busca por similaridade vetorial no PostgreSQL. Disponível em: `https://github.com/pgvector/pgvector`. Acesso em: 22 jun. 2026.

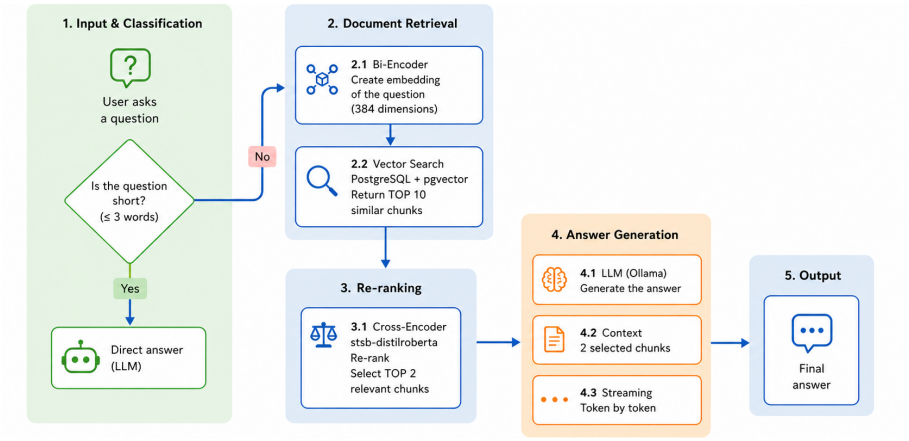


Figura 2. Pipeline de geração da resposta.

(Ollama)) a um LLM executado localmente via Ollama, que gera a resposta final em português, transmitida via subetapa 4.3 (*Streaming*), resultando na etapa 5 (*Output*) com a resposta final.

3.2. Arquitetura Geral, v2

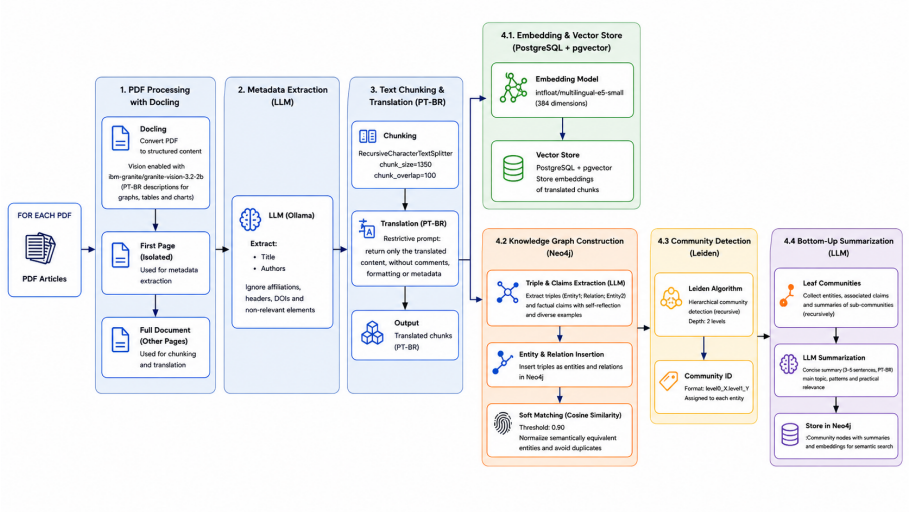


Figura 3. Pipeline de ingestão de documentos da versão híbrida.

A Figura 3 apresenta o *pipeline* de ingestão da versão híbrida do sistema. Esta arquitetura evolui a versão anterior ao introduzir capacidades multimídia na extração de dados, um novo modelo de *embeddings* e a construção paralela de um grafo de conhecimento no *Neo4j*[5], estruturado por meio de detecção hierárquica de comunidades e sumarização *bottom-up*.

A primeira modificação ocorre na etapa 1 (*PDF Processing with Docling*). Diferentemente da versão anterior, a descrição de imagens é ativada por meio do modelo *vision-language* `ibm-granite/granite-vision-3.2-2b`, que gera descrições

[5]Banco de dados orientado a grafos. Disponível em: `https://neo4j.com/`. Acesso em: 22 jun. 2026.

textuais para gráficos, figuras e tabelas, enriquecendo o conteúdo processado tanto na primeira página quanto no restante do documento. Na etapa 2 (*Metadata Extraction (LLM)*), o processo de isolar a primeira página via Ollama é refinado para que a LLM ignore explicitamente afiliações, cabeçalhos, DOIs e outros elementos não relevantes, retendo estritamente o título e os autores. A etapa 3 (*Text Chunking & Translation (PT-BR)*) modifica os parâmetros de segmentação do `RecursiveCharacterTextSplitter`, ampliando o *chunk size* para 1.350 caracteres, enquanto mantém o *overlap* em 100 caracteres. Além disso, a tradução dos *chunks* passa a utilizar um *prompt* restritivo que impede a inclusão de comentários, formatações ou metatextos pela LLM, garantindo um *output* limpo. Na etapa 4.1 (*Embedding & Vector Store (PostgreSQL + pgvector)*), o modelo anterior é substituído pelo `intfloat/multilingual-e5-small`, que possui dimensão de 384 e oferece suporte nativo ao português brasileiro para a geração dos vetores armazenados no `PostgreSQL` com a extensão `pgvector`.

As etapas seguintes são exclusivas desta segunda arquitetura. Paralelamente ao armazenamento vetorial, cada *chunk* é direcionado à etapa 4.2 (*Knowledge Graph Construction (Neo4j)*), iniciando-se pela subetapa de *triple & claims extraction (LLM)*, onde um *prompt* especializado com mecanismos de *self-reflection* e exemplos extrai triplas no formato (Entidade1; Relação; Entidade2) e *claims* (afirmações factuais). Na subetapa de *entity & relation insertion*, os dados são populados no *Neo4j*, aplicando-se um processo de *soft matching (cosine similarity)* com limiar de 0,90 para normalizar entidades semanticamente equivalentes e evitar duplicidade. Sobre o grafo construído, aplica-se a etapa 4.3 (*Community Detection*), que utiliza o algoritmo Leiden de forma recursiva para a detecção hierárquica de comunidades em dois níveis de profundidade. Cada entidade recebe um identificador no formato `level0_X.level1_Y`, mapeando sua posição na estrutura. Por fim, a etapa 4.4 (*Bottom-Up Summarization*) realiza a sumarização das comunidades a partir das *leaf communities* (nível mais profundo). Os dados consolidados são processados na subetapa de *LLM summarization*, gerando resumos concisos de três a cinco frases em português sobre o tema central e os padrões observados. O fluxo encerra-se na subetapa de *Store in Neo4j*, onde esses resumos são salvos como nós `:Community`, acompanhados de seus respectivos *embeddings* vetoriais para habilitar a busca semântica na fase de consulta.

3.3. Recuperação de informação

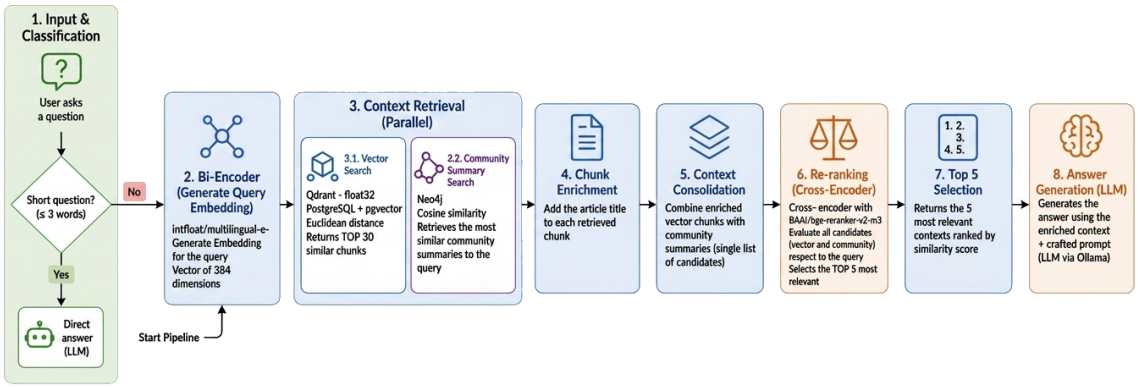


Figura 4. Pipeline de geração de resposta para a abordagem híbrida.

A Figura 4 ilustra o *pipeline* de geração de resposta da versão híbrida do sistema.

Esta arquitetura evolui o modelo anterior ao introduzir uma estratégia de recuperação paralela que combina a busca vetorial tradicional com a busca por resumos de comunidades estruturadas no grafo, além de atualizar os modelos de *embedding* e de ordenação.

O fluxo inicia-se na etapa 1 (*Entrada e Classificação*), mas altera-se na etapa 2 (*Bi-Encoder (Geração do Embedding da Pergunta)*), onde a pergunta do usuário passa a ser codificada pelo modelo `intfloat/multilingual-e5-small`, substituindo o modelo de *embeddings* anterior.

A principal mudança estrutural ocorre na etapa 3 (*Recuperação de Contextos (Paralelo)*). Enquanto a subetapa 3.1 (*Busca Vetorial*) mantém a busca por similaridade euclidiana no banco de dados vetorial, um processo concorrente é disparado na subetapa 3.2 (*Busca por Resumos de Comunidades*). Nesta subetapa, o sistema realiza uma busca vetorial baseada em similaridade cosseno diretamente nos resumos de comunidades armazenados no *Neo4j*. Antes do processo de ordenação final, são adicionadas duas etapas exclusivas desta arquitetura. Na etapa 4 (*Enriquecimento dos Chunks*), cada *chunk* recuperado recebe o metadado do título do artigo de origem para que a LLM identifique as fontes. Em seguida, na etapa 5 (*Consolidação dos Contextos*), os candidatos de ambas as fontes (*chunks* vetoriais e resumos do grafo) são unificados em uma lista única. O processo de filtragem também foi modificado na etapa 6 (*Re-ranqueamento (Cross-Encoder)*), que passa a utilizar o modelo `BAAI/bge-reranker-v2-m3`. Esse *cross-encoder* avalia a relevância global da lista unificada, sem distinção de origem dos contextos. Na etapa 7 (*Seleção dos Top 5*), o critério de corte é expandido para selecionar os cinco contextos mais relevantes, em contraste com os dois utilizados na primeira versão. Por fim, na etapa 8 (*Geração da Resposta (LLM)*), esse bloco consolidado e enriquecido é injetado no *prompt* final para que a LLM gere a resposta em linguagem natural.

4. Avaliação

A avaliação das arquiteturas propostas foi conduzida por meio de um conjunto de 22 perguntas, cada uma acompanhada de sua respectiva resposta esperada (*ground truth*). Essas perguntas contemplaram os cinco artigos processados. Importante destacar que essas perguntas foram criadas pelos próprios autores de cada trabalho, sendo cinco perguntas para o artigo de [da Silva Ferreira et al. 2013], cinco para [Lopes et al. 2013], seis para [Pontes et al. 2019], três para [Cavalcante et al. 2019] e três para [de Bodas Terassi et al. 2023]. Foram escolhidos esses artigos, pois eles representam uma diversidade de tópicos para o objeto de estudo, além de que os autores desses artigos ainda estão atuando como colaboradores do ITV no momento de escrita deste artigo.

O processo experimental utilizou o modelo `gpt-oss-20b` para inferência sobre o conjunto de perguntas. As respostas foram avaliadas via abordagem *LLM-as-a-judge* utilizando o *framework* DeepEval[6]. Sete modelos atuaram como juízes na avaliação das métricas: `qwen3:32b`, `qwen3:8b`, `llama3.3:70b`, `gemma3:27b`, `gemma3:12b`, `mixtral:8x22b` e `mixtral:8x7b`.

As respostas geradas foram avaliadas com base nas seguintes métricas nativas do *framework*: *correctness* (correção), *answer relevance* (relevância da resposta), *faithfulness* (fidelidade ao contexto), *contextual relevance* (relevância contextual) e *contextual*

[6] `https://www.deepeval.com/`

recall (capacidade de recuperar informações relevantes do contexto). Para mitigar a variabilidade inerente às respostas dos LLMs e garantir a robustez estatística dos resultados, os experimentos foram executados múltiplas vezes, com uma frequência de repetição entre 5 e 10 execuções por cenário avaliado. Os testes foram integralmente processados em um ambiente local dedicado, cujas especificações de *hardware* e *software* compreendem: um processador Intel Core i7-14700KF, memória RAM de 64 GB DDR5, armazenamento em SSD de 1 TB, e uma GPU NVIDIA GeForce RTX 3090 com 24 GB de VRAM, operando sob o sistema operacional Ubuntu 24.04.3 LTS. O tempo total estimado para a conclusão de todo o processo de ingestão dos dados e julgamentos foi de aproximadamente 7 horas.

5. Resultados

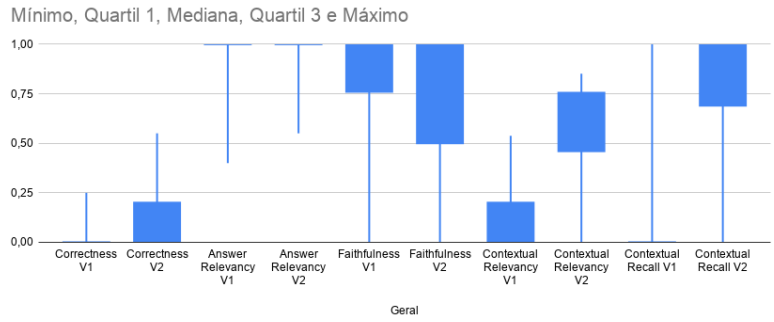


Figura 5. Análise comparativa das métricas de avaliação do pipeline RAG entre as versões V1 e V2 via diagramas de caixa (Boxplot), mapeando a distribuição estatística de mínimo, primeiro quartil (Q_1), mediana, terceiro quartil (Q_3) e máximo.

A Figura 5 mostra que a versão V2 superou o gargalo de busca da V1, apresentando melhora expressiva em *Contextual Recall* e *Contextual Relevancy*. Esse refinamento reduziu a omissão de conhecimento e elevou a exatidão da geração (*Correctness*), mantendo estáveis os índices de *Faithfulness* e *Answer Relevance*. Adicionalmente, a Figura 6 detalha a variabilidade e a consistência dos sete modelos julgadores por meio de *boxplots*, permitindo uma análise diagnóstica mais robusta das arquiteturas propostas.

Análise Agregada: Robustez e Confiabilidade. Os resultados agregados confirmam ganhos substanciais do modelo híbrido, mas um achado igualmente importante emerge da análise detalhada: a **variabilidade das avaliações reduz drasticamente** na transição de V1 para V2. Enquanto o RAG Vetorial apresenta *boxplots* com *spreads* amplos (indicando desacordo entre julgadores), o modelo híbrido exibe *boxplots* comprimidos, especialmente para *contextual recall*, onde todos os sete modelos concordam em medianas entre 0,75 e 0,95. Isso indica que o *GraphRAG* não apenas melhora a performance média, mas produz recuperações mais **previsíveis e confiáveis**, um aspecto crítico para sistemas de produção. A *contextual relevance* mantém padrão similar: V1 varia de 0,20 a 0,35, enquanto V2 concentra-se entre 0,60 e 0,70 em quase todos os julgadores.

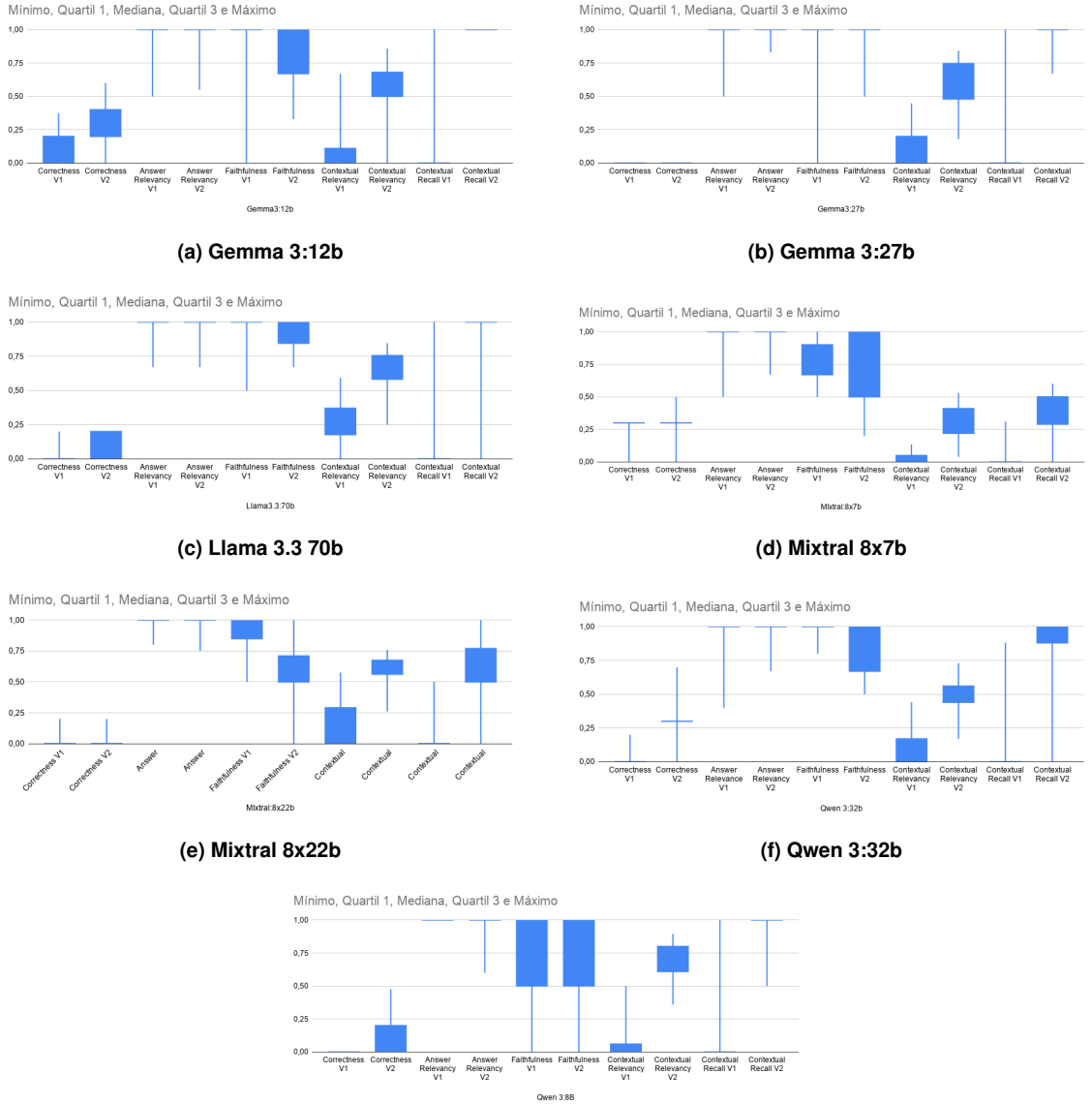
Divergências entre Modelos Julgadores. A análise individual dos modelos revela heterogeneidade nas avaliações que fornece pistas sobre as características das arquiteturas.

O **Gemma 3:12b** (Figura 6-(a)) avalia com critérios particularmente rigorosos, apresentando *boxplots* dos mais apertados da bateria, porém suas medianas para *contextual recall* em V2 atingem aproximadamente 0,95—a mais alta entre todos. Isso sugere que modelos menores, quando calibrados, podem ser mais discriminativos. Em contraste, o **Llama 3.3 70b** (Figura 6-(c)) expõe uma fraqueza crítica do RAG vetorial: seu *boxplot* de *contextual recall* V1 apresenta um *outlier* elevado ($\sim$0,6) acompanhado de mediana baixa ($\sim$0,15), indicando que o modelo maior ocasionalmente captura recuperações adequadas, mas falha sistematicamente. Crucialmente, V2 **elimina essa variabilidade**, com mediana $\sim$0,90 e *spread* mínimo, sugerindo que o *GraphRAG* estabiliza a recuperação mesmo sob diferentes tipos de perguntas. O **Mixtral 8x22b** (Figura 6-(e)) diferencia-se por exibir a maior variabilidade em *faithfulness*, com *spread* de aproximadamente 0,30 mesmo em V2 (variando de 0,6 a 1,0). Enquanto outros modelos concordam que *faithfulness* cai moderadamente de 0,90 para 0,80, Mixtral detecta inconsistências maiores. Interpretação: ao expandir de 2 para 5 *chunks*, algumas respostas introduzem nuances que podem estar parcialmente ancoradas ao contexto, revelando um *trade-off* entre completude contextual e fidelidade estrita—uma característica do design que merece atenção.

O Paradoxo da Correctness e a Validação do Ganho. Um padrão emerge ao observar a métrica de *correctness* nos *boxplots*: todos os sete modelos julgadores concordam que essa métrica permanece problemática (medianas abaixo de 0,40 mesmo em V2), apesar da melhoria em recuperação de contexto. O **Qwen 3:32b** (Figura 6-(f)) exemplifica essa validação: sua progressão é uniforme e equilibrada (tanto para V1 quanto V2), refletindo os ganhos agregados sem distorções. Seus *boxplots* de *answer relevance* permanecem superapertados em 0,98 para ambas as versões, evidenciando que a melhoria é robusta, e não um artefato de um avaliador em específico. Esses resultados demonstram que a abordagem híbrida, ao integrar busca vetorial, grafo de conhecimento e *reranking*, tende a melhorar significativamente a recuperação de contexto, e também **reduz a variabilidade intermodelo** e aumenta a previsibilidade das avaliações. Porém, a persistência de *correctness* baixa sugere que mecanismos adicionais de verificação factual pós-geração são necessários para sistemas que demandem alta fidelidade factual; uma possivel direção para trabalhos futuros.

6. Limitações

A capacidade de generalização dos resultados obtidos é limitada pelo tamanho reduzido e pelo escopo altamente específico do *dataset*, que conta com apenas 22 perguntas restritas ao domínio da climatologia da Amazônia, além de estar sujeito ao potencial viés decorrente de sua seleção manual. Adicionalmente, a adoção da abordagem *LLM-as-a-judge* introduz uma dependência direta da qualidade dos modelos avaliadores e de seus respectivos vieses inerentes, o que torna recomendável a inclusão de uma validação por especialistas em trabalhos futuros. Por fim, a execução dos experimentos em um ambiente local e controlado via Ollama, somada ao ajuste puramente empírico dos hiperparâmetros, impõe restrições computacionais e operacionais, exigindo validações adicionais antes que esses resultados possam ser extrapolados para cenários de produção ou aplicados a novos contextos.



(a) Gemma 3:12b

(b) Gemma 3:27b

(c) Llama 3.3 70b

(d) Mixtral 8x7b

(e) Mixtral 8x22b

(f) Qwen 3:32b

(g) Qwen 3:8b

Figura 6. Boxplots de avaliação para todos os modelos julgadores.

7. Trabalhos Relacionados

O trabalho de [Han et al. 2025] fornece a comparação mais direta com a nossa abordagem ao avaliar sistematicamente RAG vetorial e variantes de *GraphRAG* em tarefas de QA (*single* e *multi-hop*) e sumarização nos domínios da Wikipedia, notícias e literatura. Os autores concluíram que as tecnologias são complementares: o RAG destaca-se em consultas factuais locais, enquanto o *GraphRAG* é superior em raciocínios complexos. Em contraste com esse foco em *benchmarks* gerais, o presente artigo aplica uma arquitetura híbrida ao domínio da climatologia, validando o uso de detecção de comunidades Leiden e triplas semânticas para integrar dados científicos dispersos.

No que tange a infraestruturas e métricas, três estudos propõem métodos que contrastam ou respaldam as escolhas desta pesquisa. [Laban et al.] introduziram o *benchmark* SummHay para avaliar a sumarização de múltiplos documentos extensos, revelando

que os sistemas RAG avançados ainda perdem para humanos em cobertura de *insights*. Esse cenário corrobora nossa escolha pela sumarização *bottom-up* de comunidades para sintetizar múltiplos artigos.

[Pradeep et al. 2025] propuseram o *AutoNuggetizer*, que automatiza a avaliação baseada em fatos atômicos (*nuggets*), demonstrando forte correlação entre o julgamento de LLMs e o de especialistas humanos no *TREC 2024 RAG Track*. Essa precisão factual atômica surge como alternativa futura para refinar a métrica de *correctness* utilizada em nossa base. Por fim, no cenário de otimização em domínios específicos, [Setty et al. 2024] avaliaram o impacto de técnicas *zero-shot* (como expansão de consulta e *reranking*) via *framework* Ragas no domínio financeiro. Os resultados indicam que o *reranking* mitiga as limitações de recuperadores densos tradicionais em textos de alta complexidade. Essa conclusão valida o uso de *cross-encoders* (como o *BGE-reranker*) na arquitetura híbrida proposta neste artigo.

8. Considerações Finais

Este artigo comparou o RAG vetorial clássico a uma abordagem híbrida com *GraphRAG*. Validada com 22 perguntas complexas e múltiplos LLMs, a arquitetura híbrida superou a tradicional com ganhos de 160% em relevância contextual e 177% em *recall* contextual, integrando detalhes locais a estruturas globais do grafo e superando o isolamento de documentos.

Referências

Cavalcante, R., Pontes, P., Souza-Filho, P., and De Souza, E. (2019). Opposite effects of climate and land use changes on the annual water balance in the amazon arc of deforestation. *Water Resources Research*, 55(4):3092–3106.

da Silva Ferreira, D. B., de Souza, E. B., and de Moraes, B. C. (2013). Ciclo horário da precipitação no leste da amazônia durante o período chuvoso. *Revista Brasileira de Climatologia*, 13.

de Bodas Terassi, P. M., Pontes, P. R. M., Xavier, A. C. F., Cavalcante, R. B. L., de Oliveira Serrão, E. A., Sobral, B. S., de Oliveira-Júnior, J. F., de Melo, A. M. Q., and Baratto, J. (2023). A comprehensive analysis of regional disaggregation coefficients and intensity-duration-frequency curves for the itacaiúnas watershed in the eastern brazilian amazon: Pm de bodas terassi et al. *Theoretical and Applied Climatology*, 154(3):863–880.

Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S., Metropolitansky, D., Ness, R. O., and Larson, J. (2024). From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Han, H., Ma, L., Wang, Y., Shomer, H., Lei, Y., Qi, Z., Guo, K., Hua, Z., Long, B., Liu, H., et al. (2025). Rag vs. graphrag: A systematic evaluation and key insights. *arXiv preprint arXiv:2502.11371*.

Laban, P., Fabbri, A. R., Xiong, C., and Wu, C.-S. Summary of a haystack: a challenge to long-context llms and rag systems (2024). *URL https://arxiv. org/abs/2407.01370*.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for

knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Lopes, M. N. G., De Souza, E. B., and da Silva Ferreira, D. B. (2013). Climatologia regional da precipitação no estado do pará. *Revista Brasileira de Climatologia*, 12.

Pontes, P. R., Cavalcante, R. B., Sahoo, P. K., da Silva Júnior, R. O., da Silva, M. S., Dall'Agnol, R., and Siqueira, J. O. (2019). The role of protected and deforested areas in the hydrological processes of itacaiúnas river basin, eastern amazonia. *Journal of Environmental Management*, 235:489–499.

Pradeep, R., Thakur, N., Upadhyay, S., Campos, D., Craswell, N., Soboroff, I., Dang, H. T., and Lin, J. (2025). The great nugget recall: Automating fact extraction and rag evaluation with large language models. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 180–190.

Sequeda, J., Allemang, D., and Jacob, B. (2024). A benchmark to understand the role of knowledge graphs on large language model's accuracy for question answering on enterprise sql databases. In *Proceedings of the 7th Joint Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA)*, pages 1–12.

Setty, S., Thakkar, H., Lee, A., Chung, E., and Vidra, N. (2024). Improving retrieval for rag based question answering models on financial documents. *arXiv preprint arXiv:2404.07221*.