Research, Society and Development, v. 14, n. 10, e106141049768, 2025
(CC BY 4.0) | ISSN 2525-3409 | DOI: http://dx.doi.org/10.33448/rsd-v14i10.49768

Bancos Vetoriais e Modelos de Embedding: Avaliação comparativa de desempenho na recuperação semântica em Língua Portuguesa

Vector Databases and Embedding Models: Comparative evaluation of performance in semantic retrieval in Portuguese

Base de Datos Vectoriales y Modelos de Incrustación: Evaluación comparativa del rendimiento en la recuperación semántica en Portugués

Recebido: 09/10/2025 | Revisado: 15/10/2025 | Aceitado: 15/10/2025 | Publicado: 17/10/2025

Patrick Fernandes Rezende Ribeiro
ORCID: https://orcid.org/0000-0002-5973-1110
Universidade Federal do Paraná, Brasil
E-mail: patrick.ribeiro@ufpr.br
Juliane de Lima Pires
ORCID: https://orcid.org/0009-0001-1158-8087
Universidade Federal do Paraná, Brasil
E-mail: julianepires@ufpr.br
Patrick Alves Bastos
ORCID: https://orcid.org/0000-0003-0017-0467
Universidade Federal do Paraná, Brasil
E-mail: patrickalves@ufpr.br
Roberto Rigo
ORCID: https://orcid.org/0009-0007-4634-8298
Universidade Federal do Paraná, Brasil
E-mail: robertorigo@ufpr.br
Matheus Henrique Assumpção dos Reis
ORCID: https://orcid.org/0009-0008-3996-8824
Universidade Federal do Paraná, Brasil
E-mail: assumpcao@ufpr.br
Kamilly Voitkiv Hubner
ORCID: https://orcid.org/0009-0006-1715-9677
Universidade Federal do Paraná, Brasil
E-mail: kamillyhubner@ufpr.br
Maria Fernanda Zandoná Casagrande
ORCID: https://orcid.org/0009-0004-9972-1752
Universidade Federal do Paraná, Brasil
E-mail: maria.fernandazandona@ufpr.br
Bruno de Paula Marafiga
ORCID: https://orcid.org/0009-0002-0806-7771
Universidade Federal do Paraná, Brasil
E-mail: brunomarafiga@ufpr.br
Dante Krol Simba
ORCID: https://orcid.org/0009-0003-4527-972X
Universidade Federal do Paraná, Brasil
E-mail: simba.dante@ufpr.br
Denise Fukumi Tsunoda
ORCID: https://orcid.org/0000-0002-5663-4534
Universidade Federal do Paraná, Brasil
E-mail: dtsunoda@ufpr.br

Resumo

O crescimento do uso de modelos de linguagem de grande escala intensificou a demanda por bancos de dados vetoriais capazes de lidar com representações semânticas de alta dimensionalidade. Este estudo teve como objetivo avaliar comparativamente diferentes combinações entre bancos de dados vetoriais e modelos de *embedding* multilíngues, considerando sua aplicabilidade à recuperação semântica em língua portuguesa. A pesquisa caracteriza-se como experimental e aplicada, conduzida em ambiente local, estruturada em quatro etapas: construção da base de dados, definição de critérios de seleção, implementação de um pipeline de experimentação e realização de avaliações de relevância, diversidade e eficiência. Foram analisadas métricas clássicas de recuperação de informação (Recall@k e nDCG), além de métricas de diversidade e equilíbrio (α-nDCG e ILD) e indicadores de eficiência computacional

(latência média, latência p95, uso médio de CPU, uso de RAM e Queries per Second - QPS). Os resultados mostraram que soluções como Milvus e Weaviate se destacam em cenários de maior demanda computacional, enquanto pgvector se mostrou mais eficiente em termos de memória. Alternativas como Chroma e pgvector, demonstraram viabilidade em contextos de menor escala. Entre os modelos de *embedding*, observou-se desempenho consistente dos modelos multilíngues disponíveis no Hugging Face para tarefas em português. Como contribuição, este trabalho apresenta uma análise empírica sistemática que evidencia as potencialidades e limitações de combinações banco vetorial/*embedding*, oferecendo subsídios para decisões práticas em projetos de curadoria digital, observatórios de dados e sistemas de recomendação em língua portuguesa.
Palavras-chave: Bancos de Dados Vetoriais; Modelos de *Embedding*; Recuperação Semântica; Avaliação de Desempenho; Língua Portuguesa.

Abstract
The growth in the use of large-scale language models has intensified the demand for vector databases capable of handling high-dimensional semantic representations. This study aimed to comparatively evaluate different combinations of vector databases and multilingual embedding models, considering their applicability to semantic retrieval in the Portuguese language. The research is characterized as experimental and applied, conducted in a local environment, and structured in four stages: database construction, definition of selection criteria, implementation of an experimentation pipeline, and evaluation of relevance, diversity, and efficiency. Classic information retrieval metrics (Recall@k and nDCG) were analyzed, in addition to diversity and balance metrics (α-nDCG and ILD) and computational efficiency indicators (average latency, p95 latency, average CPU usage, RAM usage, and Queries per Second - QPS). The results showed that solutions such as Milvus and Weaviate stand out in scenarios with higher computational demand, while pgvector proved to be more efficient in terms of memory. Alternatives such as Chroma and pgvector demonstrated viability in smaller-scale contexts. Among the embedding models, consistent performance was observed in the multilingual models available on Hugging Face for tasks in Portuguese. As a contribution, this work presents a systematic empirical analysis that highlights the potential and limitations of different vector bank/embedding combinations, offering support for practical decisions in digital curation projects, data observatories, and recommendation systems in Portuguese.
Keywords: Vector Databases; Embedding Models; Semantic Retrieval; Performance Evaluation; Portuguese Language.

Resumen
El crecimiento en el uso de modelos de lenguaje de gran tamaño ha intensificado la demanda de base de datos vectoriales capaces de manejar representaciones semánticas de alta dimensión. Este estudio tuvo como objetivo fue evaluar comparativamente diferentes combinaciones de base de datos vectoriales y modelos de incrustación multilingües, considerando su aplicabilidad de la recuperación semántica en el idioma portugués. La investigación es experimental y aplicada, realizada en un entorno local y estructurada en cuatro etapas: construcción de la base de datos, definición de los criterios de selección, implementación del experimentación y evaluación de relevancia, diversidad y eficiencia. Se analizaron las métricas clásicas de recuperación de información (Recall@k y nDCG), métricas de diversidad y equilibrio (α-nDCG e ILD) e indicadores de eficiencia computacional (latencia media, latencia p95, uso medio de la CPU, uso de la RAM y consultas por segundo – QPS). Los resultados mostraron que Milvus y Weaviate destacan en escenario con mayor demanda computacional, mientras que pgvector demostró ser más eficiente en términos de memoria. Alternativas como Chroma y pgvector demostraron su viabilidad en contextos a menor escala. Entre los modelos de incrustación, se observó un rendimiento consistente en los modelos multilingües disponibles no Hugging Face para tareas en portugués. Como contribución, este trabajo presenta un análisis empírico sistemático que destaca el potencial y las limitaciones de diferentes combinaciones de bancos de vectores/incrustaciones, ofreciendo apoyo para la toma de decisiones prácticas en proyectos de conservación digital, observatorios de datos y sistemas de recomendación en portugués
Palabras clave: Bases de Datos Vectoriales; Modelos de Incrustaciones; Recuperación Semántica; Evaluación de Rendimiento; Lengua Portuguesa.

1. Introdução

O volume crescente de dados digitais, juntamente com os avanços recentes em inteligência artificial, intensificou a necessidade de métodos eficientes para a recuperação de informação. Nesse cenário, os modelos de linguagem e as técnicas de representação vetorial passaram a ocupar papel central em aplicações como sistemas de busca e recomendação, permitindo análises semânticas mais precisas e contextualizadas. Os bancos de dados vetoriais, por sua vez, constituem a infraestrutura necessária para armazenar e processar *embeddings* de alta dimensionalidade, o que os torna componentes estratégicos no desenvolvimento de soluções baseadas em inteligência artificial.

Apesar da relevância do tema, observa-se escassez de estudos comparativos que analisem o desempenho desses bancos de dados em língua portuguesa. A maior parte dos *benchmarks* disponíveis está voltada ao inglês, dificultando a avaliação crítica das soluções em contextos nacionais. Essa lacuna cria obstáculos para profissionais e pesquisadores que precisam selecionar tecnologias adequadas para projetos de ciência de dados e sistemas de informação em português.

A discussão é pertinente não apenas no âmbito técnico, mas também nos impactos sociais e institucionais. Setores como educação, saúde, turismo e segurança pública dependem de sistemas capazes de recuperar informação de forma eficiente, confiável e adaptada à realidade linguística local. Assim, a escolha do banco de dados vetorial pode impactar na qualidade dos serviços prestados e na tomada de decisão baseada em dados.

Este artigo avalia comparativamente a combinação entre diferentes bancos de dados vetoriais e modelos de *embedding* no desempenho de tarefas de recuperação semântica, mensurado por precisão de recuperação, tempo de resposta e eficiência, em sistemas de recomendação baseados em busca por similaridade para o idioma português. Com isso, visa-se responder à seguinte pergunta de pesquisa: "Qual é o impacto da combinação entre diferentes bancos de dados vetoriais e modelos de *embedding* no desempenho de tarefas de recuperação semântica por busca de similaridade para o idioma português?". Assim, este estudo teve como objetivo geral avaliar comparativamente combinações entre bancos de dados vetoriais e modelos de *embedding* multilíngues, considerando sua aplicabilidade à recuperação semântica em língua portuguesa.

2. Metodologia

A pesquisa se configura como um estudo experimental de caráter empírico aplicado, orientado pela busca de evidências obtidas de forma sistemática e controlada (Kerlinger, 1980). Com o propósito central de avaliar comparativamente combinações entre bancos de dados vetoriais e modelos de *embedding* no desempenho de tarefas de recuperação semântica, verifica ainda a existência de diferenças estatisticamente significativas entre essas combinações na execução de buscas por similaridade.

O período de coleta, construção, experimentação e avaliação estendeu-se de primeiro de março a quatro de agosto de 2025, contemplando implementações em ambiente local. Como limitações metodológicas, definiu-se o escopo do estudo para: banco de dados vetoriais gratuitos e de código aberto, modelos de *embedding* disponíveis na plataforma *Hugging Face*, avaliação restrita a métricas de similaridade e desempenho e a utilização de uma base de dados específica do domínio de ferramentas de Inteligência Artificial (IA).

O delineamento metodológico foi estruturado em quatro etapas principais. Na primeira, realizou-se a construção da base de dados com apoio de ferramentas de inteligência artificial. Em seguida, foram definidos os critérios de seleção tanto para os bancos de dados vetoriais quanto para os modelos de *embedding*. A terceira etapa consistiu na elaboração de um pipeline experimental que possibilitou a indexação, a execução de consultas e a coleta das métricas de avaliação.

Por fim, as combinações selecionadas foram submetidas a dois tipos de análise: (i) avaliação de relevância e diversidade, mensurada por meio de Recall@k, nDCG@k, α-nDCG, ILD; e (ii) avaliação da eficiência computacional, que incluiu a execução de consultas reais com mensuração da latência média, latência p95, *Queries per Second* (QPS) e utilização de recursos computacionais CPU e memória RAM.

3. Resultados

Esta seção apresenta os resultados e a fundamentação teórica que sustentam as análises experimentais realizadas. Inicialmente, são abordados conceitos e estudos que subsidiam o uso de bancos de dados vetoriais e modelos de *embedding* (Seção 3.1). Em seguida, descrevem-se o fluxo de construção e avaliação do experimento (Seção 3.2), bem como os resultados empíricos e comparativos obtidos nas análises de relevância, diversidade e eficiência computacional.

3.1 Contextualização conceitual

Os bancos de dados vetoriais surgem como uma resposta às limitações dos sistemas relacionais tradicionais no tratamento de dados de alta dimensionalidade, típicos de aplicações em inteligência artificial e aprendizado de máquina. Esses sistemas são projetados para armazenar, indexar e recuperar vetores de grande dimensionalidade, possibilitando pesquisas por similaridade com maior eficiência (Joshi, 2025).

Diferentemente dos bancos relacionais, que estruturam dados em tabelas, os bancos vetoriais trabalham com *embeddings* — representações numéricas de textos, imagens ou sinais — que capturam relações semânticas entre os objetos. Esse formato permite operações de similaridade baseadas em métricas como a distância euclidiana e a similaridade do cosseno, fundamentais para aplicações de busca semântica, sistemas de recomendação e modelos generativos (Zhang; Liu; Wang, 2024).

A eficiência desses sistemas depende da escolha de algoritmos de indexação apropriados. Entre os mais utilizados destacam-se:

- Árvores de Partição Aleatória (*Random Projection Trees*);
- *Hierarchical Navigable Small World Graphs* (HNSW);
- *Product Quantization* (PQ).

Esses métodos possibilitam buscas aproximadas em grandes volumes de dados, reduzindo o custo computacional e mantendo precisão aceitável, especialmente em cenários de escalabilidade (Latimer, 2024).

Enquanto bancos relacionais como PostgreSQL, com extensões como pgvector, conseguem oferecer suporte limitado a vetores, apresentam problemas de desempenho e escalabilidade ao lidar com dados de alta dimensionalidade. Estudos recentes demonstram que, embora funcionais, tais soluções não são otimizadas para consultas por similaridade, tornando os bancos vetoriais especializados mais adequados para aplicações nas quais a velocidade e a precisão na recuperação são críticas (Zhang; Liu; Wang, 2024; Srivastava, 2023).

Dessa forma, observa-se que os bancos de dados vetoriais não somente complementam os sistemas relacionais tradicionais, mas se consolidam como infraestrutura indispensável em aplicações que demandam recuperação semântica. A evolução dessas tecnologias está diretamente relacionada ao desenvolvimento de modelos de linguagem de grande escala, que dependem da representação vetorial para processar dados de maneira contextualizada.

Modelos de *Embedding* Semântico

O desenvolvimento dos Modelos de Linguagem de Grande Escala (LLMs) ampliou significativamente as possibilidades de aplicação dos bancos de dados vetoriais. Esses modelos produzem representações numéricas de textos, denominadas *embeddings*, que sintetizam relações semânticas complexas entre palavras, frases ou documentos. Para tais representações serem utilizadas eficientemente, adotam-se mecanismos capazes de armazenar e recuperar vetores em ambientes de alta dimensionalidade, papel desempenhado pelos bancos vetoriais (Lewis *et al.,* 2020).

Entre as arquiteturas que se beneficiam dessa integração destaca-se o *Retrieval-Augmented Generation* (RAG), que combina busca vetorial e geração de linguagem natural. O processo ocorre em duas etapas: na primeira, o sistema realiza a recuperação de vetores semanticamente próximos em um banco de dados vetorial; na segunda, o modelo de linguagem utiliza o material recuperado como contexto para a elaboração da resposta. Esse procedimento contribui para reduzir limitações típicas dos modelos de geração, a exemplo de produção de respostas imprecisas ou desatualizadas, além de aumentar a confiabilidade da informação apresentada (Joshi, 2025).

No entanto, o uso de bancos relacionais em cenários de integração com LLMs mostra-se restrito, em virtude do custo computacional associado a consultas em espaços de alta dimensionalidade. Por esse motivo, soluções especializadas, como

Milvus, Weaviate, Qdrant e Pinecone, vêm sendo amplamente utilizadas. Esses sistemas oferecem integração nativa com modelos de linguagem e suporte a consultas em tempo real, o que possibilita maior escalabilidade e precisão em tarefas de recuperação semântica (Latimer, 2024; Srivastava, 2023).

Sistemas de Recomendação Semântica

A avaliação constitui etapa essencial em qualquer proposta científica ou tecnológica, por ser por meio dela que se verifica a validade dos métodos empregados e a relevância dos resultados alcançados. No campo dos sistemas de informação, esse processo é indispensável para assegurar que os modelos desenvolvidos atendam aos objetivos estabelecidos e ofereçam resultados de qualidade. No caso específico da recuperação de informação, a etapa de avaliação é ainda mais relevante, já que a utilidade prática do sistema depende diretamente da pertinência, da cobertura e da organização dos itens recuperados.

Para mensurar o desempenho em tarefas de busca semântica, diversas métricas vêm sendo consolidadas na literatura. Entre elas, destacam-se Precision@k e Recall@k, que avaliam, respectivamente, a proporção de itens relevantes entre os k primeiros resultados e a capacidade do sistema de recuperar todos os itens relevantes disponíveis. Essas duas medidas são complementares, uma vez que a precisão privilegia a qualidade imediata dos resultados apresentados, enquanto a revocação se concentra na abrangência da recuperação (Manning; Raghavan; Schütze, 2008).

Outra métrica amplamente utilizada é o *Normalized Discounted Cumulative Gain* (nDCG), que introduz a dimensão da ordenação no processo de avaliação. Nessa métrica, itens mais relevantes recebem maior peso quando aparecem nas primeiras posições do ranqueamento, de modo que a hierarquia dos resultados influencia diretamente a pontuação final do sistema. Essa característica torna o nDCG especialmente adequado para aplicações nas quais a posição do item recuperado impacta a experiência do usuário (Järvelin; Kekäläinen, 2002).

Além disso, a métrica *Maximal Marginal Relevance* (MMR) busca equilibrar relevância e diversidade, reduzindo a ocorrência de resultados redundantes e promovendo a inclusão de informações complementares. Essa abordagem é particularmente importante em sistemas de recomendação e em arquiteturas baseadas em *Retrieval-Augmented Generation* (RAG), nas quais a variedade de informações recuperadas contribui para respostas mais completas e contextualizadas (Carbonell; Goldstein, 1998).

As mencionadas métricas permitem uma análise abrangente e crítica do desempenho de sistemas de recuperação de informação, fornecendo subsídios tanto para comparações experimentais entre diferentes bancos de dados vetoriais quanto para decisões práticas de adoção em ambientes organizacionais e acadêmicos.

Trabalhos relacionados

Os bancos de dados vetoriais viabilizam algumas aplicações em inteligência artificial, especialmente em tarefas de busca semântica e recomendação. Estudos recentes têm explorado o desempenho de diferentes estruturas de indexação, como os *Hierarchical Navigable Small World Graphs* (HNSW) e o *Product Quantization* (PQ), reconhecidos pela eficiência em consultas aproximadas em ambientes de alta dimensionalidade (Malkov; Yashunin, 2018). Pesquisas também destacam a integração dessas bases com modelos de linguagem de grande escala, possibilitando arquiteturas como o *Retrieval-Augmented Generation* (RAG), nas quais a recuperação de informações é utilizada para enriquecer a geração de respostas (Lewis *et al.,* 2020).

Na prática, diversas soluções comerciais e de código aberto têm sido avaliadas pela literatura, a exemplo de Milvus, Weaviate, Pinecone e Qdrant, com análises voltadas à escalabilidade, flexibilidade de integração e custo de implementação (Latimer, 2024; Srivastava, 2023). Além disso, levantamentos abrangentes reforçam a relevância dos bancos vetoriais no

ecossistema contemporâneo de dados, apontando avanços, desafios e tendências para o seu desenvolvimento (Ma *et al.,* 2023; Pan *et al.,* 2024).

Embora a produção internacional seja significativa, observa-se que grande parte dos estudos se concentra em *benchmarks* realizados em inglês, o que limita a aplicabilidade dos resultados em contextos locais. Em língua portuguesa, até onde se sabe, os esforços concentram-se sobretudo em pesquisas sobre *embeddings* e representações semânticas, com foco em tarefas de similaridade textual e classificação (Hartmann *et al.,* 2017; Silva; Caseli, 2021; Souza; Santos Filho, 2022). Trabalhos específicos, como o corpus Regis, aplicado a documentos de geociências (Oliveira *et al.,* 2021), e o *dataset* JurisTCU, voltado à recuperação de informações jurídicas em português (Fernandes *et al.,* 2025), representam avanços importantes, mas ainda pontuais. Pesquisas mais recentes, como a avaliação de representações textuais para similaridade semântica em domínios jurídicos (Carvalho *et al.,* 2025), reforçam a relevância da temática, mas também evidenciam a ausência de estudos sistemáticos comparando bancos de dados vetoriais em cenários lusófonos.

Assim, identifica-se uma lacuna relevante na literatura: apesar da consolidação conceitual e do desenvolvimento de ferramentas voltadas ao armazenamento e à recuperação vetorial, ainda são escassos os trabalhos que investigam o desempenho de diferentes bancos em língua portuguesa, sobretudo considerando métricas amplamente reconhecidas em recuperação de informação, como precisão, revocação, nDCG e MMR. O presente estudo busca contribuir para minimizar essa lacuna, oferecendo uma análise comparativa aplicada à recuperação semântica em português e discutindo suas implicações práticas em diferentes áreas do conhecimento.

Dessa forma, a revisão teórica e empírica demonstra a carência de estudos comparativos envolvendo bancos vetoriais em língua portuguesa, especialmente com foco em métricas de recuperação semântica e eficiência computacional. A próxima subseção descreve o fluxo de construção e avaliação do experimento proposto, que busca preencher essa lacuna.

3.2 Fluxo de construção, seleção e avaliação

Construção da Base de dados

A construção da base de dados envolveu uma equipe multidisciplinar com integrantes das graduações em Ciência da Computação, Gestão da Informação, Tecnologia e Desenvolvimento de Sistemas e pós-graduação em Gestão da Informação.

A base de dados[1] construída encontra-se organizada e armazenada em formato .XLSX, composta por 29 atributos (colunas) e 162 registros (tuplas) referentes ao cadastro de ferramentas de inteligência artificial. Cada registro reúne informações como: nome da ferramenta, categoria, finalidade principal, principais funcionalidades, perfil de usuário, exigência ou não de programação, tipo de plataforma, idiomas disponíveis, versões, limitações da versão gratuita, planos pagos, exemplos de aplicação prática, restrições gerais, conceitos-chave associados e URL de acesso.

Além desses dados descritivos, a base inclui notas de 0 a 5 para os seguintes critérios: precisão, facilidade de uso, integração, customização, desempenho/escala, transparência/explicabilidade, custo-benefício, suporte/documentação, segurança/privacidade, inovação e acessibilidade. Para cada ferramenta, há também comentários textuais, nos quais são registradas experiências de uso e/ou opiniões a seu respeito.

Critérios de seleção

Para a seleção tanto dos bancos vetoriais quanto dos modelos de *embedding* foram definidos critérios de inclusão e exclusão. No Quadro 1, descrevem-se os critérios seguidos na seleção dos bancos de dados vetoriais.

[1] Link de acesso a planilha com a base de dados criada: https://doi.org/10.5281/zenodo.17345247

Quadro 1: Critérios de inclusão e exclusão para seleção de banco de dados vetoriais.

Critérios de Inclusão	Critérios de Exclusão
I - Possui biblioteca Python	I - Incompatibilidade com a linguagem Python
II - Suporte a *Retrieval-Augmented Generation* (RAG), técnica que combina busca e geração de texto	II -Não há suporte a *Retrieval-Augmented Generation* (RAG), técnica que combina busca e geração de texto
III - Adequação para sistemas de recomendação	III - Inadequação para Sistemas de Recomendação
IV - Capacidade de filtrar resultados com base em metadados	IV - Incapacidade de filtrar resultados com base em metadados.
V - Licenciamento *Open Source*	V - Licenciamento Não *Open Source*
VI - Suporte à persistência local	VI - Não há suporte à Persistência Local
VII - Suporte para buscas aproximadas de vizinhos mais próximos	VII - Não há suporte para buscas aproximadas de vizinhos mais próximos
VIII - Capacidade de processar consultas por segundo (*queries per second*)	VIII - Não capacidade de processar consultas por segundo (*queries per second*)
IX - Suporte para busca híbrida, que combina busca vetorial e por palavras-chave	XIX - Não há suporte para busca híbrida, que combina busca vetorial e por palavras-chave
X - É um banco vetorial dedicado otimizados para lidar com *embedding*	X - Não é um banco vetorial dedicado
XI - Suporte a filtros e customizações nativas	XI - Ausência de algoritmo baseado em hashing para buscas aproximadas de vizinhos mais próximos
XII - Busca por similaridade entre vetores	XII - Não há busca por similaridade entre vetores
XIII - Busca híbrida que combina vetorial e textual	XIII - Não há busca híbrida que combina vetorial e textual
XIV - Integração com modelos de linguagem grande (LLMs)	XIV - Não há integração com modelos de linguagem grande (LLMs)
XV - Integração direta com modelos da Hugging Face	XV - Não há integração direta com modelos da Hugging Face
XVI - Suporte à biblioteca LangChain para cadeia de ferramentas de LLMs	XVI - Não há suporte à biblioteca LangChain para cadeia de ferramentas de LLMs
XVII - Suporte à biblioteca LlamaIndex para índice e análise de dados em LLMs	XVII - Não há suporte à biblioteca LlamaIndex para índice e análise de dados em LLMs

Fonte: Elaborado pelos Autores (2025).

Para a seleção dos modelos de *embedding* foram estabelecidos critérios técnicos e funcionais que guiaram a escolha das tecnologias utilizadas no experimento, como apresentado no Quadro 2.

Quadro 2: Critérios de inclusão e exclusão para seleção de modelos de *embedding*.

Critérios de Inclusão	Critérios de Exclusão
I - Compatibilidade com a linguagem *Python*	I - Incompatibilidade com a linguagem *Python*
II - Suporte à *Retrieval-Augmented Generation* (RAG)	II - Ausência de suporte à abordagem *Retrieval-Augmented Generation* (RAG)
III - Adequação para sistemas de recomendação	III - Inadequação para sistemas de recomendação
IV - Capacidade de generalização semântica (*semantic search*)	IV - Desempenho insatisfatório em busca semântica
V - Licenciamento open source	V - Licenciamento proprietário ou restritivo
VI - Suporte multilíngues	VI - Ausência de suporte multilíngue
VII - Suporte ao idioma português	VII - Ausência de suporte ao idioma português
VIII - Compatibilidade com banco vetorial	VIII - Incompatibilidade com bancos de dados vetoriais
IX - Suporte a *embeddings* densos (*dense vectors*)	IX - Incapacidade de operar com vetores densos
X - Boa performance em tarefas de NLP gerais (classificação, *clustering*, Q&A)	X - Baixo desempenho em tarefas gerais de PLN (classificação, *clustering*, Q&A)
XI - Compatibilidade e Disponibilidade no Hugging Face	XI - Incompatibilidade com a plataforma Hugging Face
XII - Originalidade e unicidade arquitetural do modelo, considerando apenas a versão mais recente quando existirem múltiplas versões do mesmo modelo-base	XII - Modelos duplicados, versões anteriores, quantizadas ou adaptadas de outro modelo-base já incluído, exceto a versão mais atual

Fonte: Elaborado pelos Autores (2025).

Construção do pipeline experimental e Avaliação

Para a terceira etapa, foi implementado um pipeline de ingestão e pré-processamento dos dados, geração de *embeddings* e indexação em cada banco vetorial selecionando.

A última etapa do experimento proposto foi conduzida em duas fases complementares: avaliação da relevância e diversidade e avaliação sistemática de desempenho.

Todos os resultados foram registrados automaticamente em um relatório[2] de log contendo todas as métricas avaliadas, permitindo análises comparativas entre os diferentes métodos de similaridade vetorial.

Na fase de descrição da avaliação da relevância e diversidade, foi implementado um *script* Python[3] responsável para realizar as consultas por similaridade semântica. A busca foi realizada com base no algoritmo de Relevância Marginal Máxima (*Maximal Marginal Relevance* – MMR), que busca equilibrar a "relevância" dos documentos em relação à consulta com a "diversidade" entre os documentos retornados. O algoritmo foi parametrizado para retornar os k = *5* resultados mais relevantes e diversos, a partir de um conjunto inicial de *fetch_k = 20* documentos recuperados com base em similaridade vetorial (*cosine similarity*). Foram realizadas 2 consultas em linguagem natural.

Após aplicar MMR com valores de λ diferentes (ex. 03, 0,5 e 0,7), foi avaliada a relevância e a diversidade dos resultados de cada consulta em cada combinação de banco de dados e modelo de *embedding*. A relevância foi mensurada como o conjunto de resultados atendia ao *ground truth* utilizando: *Recall@k* para verificar a cobertura de itens relevantes nos top-k e nDCG@k (*Normalized Discounted Cumulative Gain*) para verificar a posição e o grau de relevância dos resultados. Já a avaliação da diversidade considerou métricas como: α-nDCG (*Alpha nDCG*), extensão do nDCG que penaliza redundância e ILD (*Intra-List Diversity*) para calcular a média da dissimilaridade entre todos os pares resultados no top-k. Depois, comparar o *trade-off* relevância *em comparação à* diversidade.

No que se refere à qualidade do ranqueamento, utilizaram-se as métricas e a-nDCG. Em que primeira considera tanto a relevância quanto a posição dos documentos (Järvelin; Kekäläinen, 2002) e, a segunda introduz o parâmetro a para balancear relevância e novidade, capturando, assim, a diversidade nos resultados apresentados. Segundo Clarke et al. (2008), essa abordagem é relevante quanto se tem cenários nos quais a redundância pode comprometer a utilidade da lista de documentos recuperados.

Na segunda fase, correspondente à avaliação sistemática de eficiência computacional, foram analisados o desempenho e a precisão do banco vetorial, por meio de outro *script* em Python[4].

A base vetorial original foi replicada em três coleções distintas em cada um dos bancos vetoriais avaliados, cada uma configurada com uma métrica de similaridade diferente: similaridade do cosseno, distância euclidiana e produto escalar.

Para cada métrica, foi realizado o seguinte procedimento: 1) seleção de 100 vetores da base de consultas; 2) cálculo do *ground truth* por meio de busca exaustiva com o algoritmo *NearestNeighbors*, com k = 10; 3) execução das buscas reais, como medição do recall médio, latência média, latência p95, QPS (*Queries per Second*) e uso de recursos computacionais CPU e RAM.

Ambiente

Os experimentos foram conduzidos em ambiente local, configurado para garantir condições controladas de execução. Utilizou-se a linguagem Python, versão 3.13.5, compilada em MSC v.1929 64 bits (AMD64), em sistema operacional Windows 11 Pro (versão 25H2). O equipamento empregado apresenta arquitetura AMD64, processador Intel Core i7-1165G7

[2] Link de acesso a todos os relatórios de logs: https://doi.org/10.5281/zenodo.17345247

[3] Link de acesso ao Script: https://doi.org/10.5281/zenodo.17345247

[4] Link de aceso ao Script: https://doi.org/10.5281/zenodo.17345247

de 11ª geração, 2.80 GHz, com 8 núcleos de CPU, 16 GB de memória RAM (3200 MT/s) e GPU dedicada NVIDIA GeForce GTX 1070 com 8 GB de memória. Essa configuração possibilitou a implementação e execução do *pipeline* experimental, incluindo a indexação, consultas e mensuração das métricas de relevância, diversidade e desempenho.

Seleção de Banco de Dados Vetoriais

Para a primeira etapa, foi adotada uma abordagem comparativa e aplicada para a seleção e utilização de bancos de dados vetoriais voltados a tarefas de recuperação semântica e recomendação inteligente com suporte a Grandes Modelos de Linguagem (LLMs). O objetivo é selecionar potenciais soluções de banco de dados vetorial alinhadas aos requisitos técnicos de um sistema de recomendação inteligente. Para isso, primeiramente, foi realizado um levantamento na web dos principais bancos de dados vetoriais disponíveis no mercado. Esta pesquisa foi realizada em 18 de junho de 2025 e retornou 20 bancos de dados.

A partir desse resultado elaborou-se o Quadro 3, apresentando o nome dos bancos vetoriais com a sua categorização, sendo "Vetorial Dedicado", aqueles otimizados para lidar com *embedding* e "Com Capacidade Vetorial", aqueles foram originalmente desenvolvidos para outros tipos de dados e depois estendidos para suportar *embeddings*.

Quadro 3: Banco de dados vetoriais levantados na pesquisa web[5]

Nº	Banco de dados	Empresa	Ano de Fundação
1	Chroma	Chromatic Enterprises Inc.	2022
2	Deep Lake	Activeloop Inc.	2018
3	Elasticsearch	Elastic N.V.	2012
4	FAISS	Meta (Facebook) AI Research	2015
5	Hnswlib	NMSLIB/Open Source	2016
6	LanceDB	LanceDB	2023
7	Marqo	Marqo	2022
8	Milvus	Zilliz	2019
9	OpenSearch	OpenSearch Project/Amazon	2021
10	pgvector	PostgreSQL Global Development Group	2021
11	Pinecone	Pinecone Systems	2019
12	Qdrant	Qdrant	2021
13	Redis	Redis Ltd.	2011
14	ScaNN	Google	2020
15	SingleStoreDB	SingleStore	2011
16	txtai	NeuML/Open Source	2020
17	Usearch	Unum Cloud/Open Source	2023
18	Vald	Yahoo Japan Corporation	2019
19	Vespa	Vespa.ai	2023
20	Weaviate	Weaviate B.V.	2019

Fonte: Elaborado pelos Autores (2025).

A partir desses critérios, foi realizada uma análise comparativa de diferentes bancos vetoriais amplamente utilizados na literatura e no mercado. O Quadro 4 apresenta o resultado da avaliação, destacando quais tecnologias atendem e quais não atendem aos critérios estabelecidos, juntamente com a justificativa para a sua exclusão.

[5] Link com o Quadro 3 completo: https://doi.org/10.5281/zenodo.17345247

Quadro 4: Avaliação comparativa de bancos de dados vetoriais com base nos critérios de seleção pré-definidos[6]

Nº	Banco de Dados Vetorial	Conclusão	Justificativa de Exclusão
1	Pinecone	Excluído	V e VII
2	Weaviate	Incluído	
3	Milvus	Incluído	
4	Deep Lake	Excluído	III e XIV
5	Vespa	Excluído	III e XIV
6	FAISS	Excluído	II, III, IV, IX, X, XII, XIII, XIV, XV e XVI
7	Qdrant	Incluído	
8	Chroma	Incluído	
9	Elasticsearch	Excluído	V, X e XVII
10	pgvector	Incluído	
11	Redis	Excluído	V e XVI
12	LanceDB	Excluído	X
13	Marqo	Excluído	XVI
14	SingleStoreDB	Excluído	V e XVI
15	Vald	Excluído	I, II, III, XII e XVI
16	Hnswlib	Excluído	II, III, IV, X e XIII
17	OpenSearch	Excluído	II, III e XIV
18	ScaNN	Excluído	X e XIII
19	txtai	Incluído	
20	Usearch	Excluído	II, III, IV, X, XII, XIII, XIV, XV e XVI

Fonte: Elaborado pelos Autores (2025).

Como resultado, foram selecionados os seguintes bancos para continuidade da implementação e testes: Weaviate, Milvus, Qdrant, Chroma, pgvector e txtai.

Seleção de Modelos de *Embeddings*

A seleção dos modelos de *embedding* semântico contou primeiramente com a triagem desses modelos na plataforma *Hugging Face*. Para isso, foram aplicados os seguintes filtros: ordenação por popularidade (*Sort: Trending*); foco na tarefa de Similaridade Semântica de Sentenças (*Tasks: Sentence Similarity*); suporte à biblioteca *sentence-transformers* (*Libraries: sentence-transformers*); compatibilidade com o idioma português ou abordagem multilíngue (com português) (*Languages: Portuguese/multiligual*). A pesquisa foi realizada no dia 18 de julho de 2025 e retornou 120 modelos.

A partir desse resultado, elaborou-se um quadro, que apresenta a ordenação, o nome do modelo, a quantidade de *downloads*, a quantidade de parâmetros e a última data de atualização do modelo. O Quadro 5 traz os 10 primeiros resultados dos 120 retornados na pesquisa.

[6] Link com o Quadro 4 completo: https://doi.org/10.5281/zenodo.17345247

Quadro 5: 10 primeiros modelos retornados na pesquisa na plataforma Hugging Face[7]

Nº	Nome do Modelo	Downloads	Parâmetros
1	sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2	11.6M	0.1B
2	intfloat/multilingual-e5-small	1.74M	0.1B
3	sentence-transformers/LaBSE	1.52M	0.3B
4	intfloat/multilingual-e5-base	1.34M	0.3B
5	Alibaba-NLP/gte-multilingual-base	10.2k	0.1B
6	ibm-granite/granite-embedding-107m-multilingual	991k	0.5B
7	armand01/paraphrase-multilingual-MiniLM-L12-v2-Q6_K-GGUF	66	0.1B
8	sizrox/paraphrase-multilingual-mpnet-base-v2-Q8_0-GGUF	9	0.3B
9	ibm-granite/granite-embedding-278m-multilingual	81.6k	0.3B
10	sentence-transformers/distiluse-base-multilingual-cased-v1	511k	0.1B

Fonte: Elaborado pelos Autores (2025).

Tendo como base esses critérios, foi efetivada uma análise comparativa de diferentes modelos de *embeddings* retornados na busca na plataforma *Hugging Face*. A Quadro 6 apresenta o resultado da avaliação, destacando quais tecnologias atendem e quais não atendem aos critérios estabelecidos, juntamente com a justificativa para a sua exclusão.

Quadro 6: Avaliação comparativa dos primeiros 10 modelos retornados na pesquisa na plataforma Hugging Face[8]

Nº	Modelos de *Embedding*	Conclusão	Justificativa de Exclusão
1	sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2	Incluído	
2	intfloat/multilingual-e5-small	Excluído	XII
3	intfloat/multilingual-e5-base	Incluído	
4	Alibaba-NLP/gte-multilingual-base	Incluído	
5	ibm-granite/granite-embedding-107m-multilingual	Excluído	XII
6	sentence-transformers/LaBSE	Incluído	
7	armand01/paraphrase-multilingual-MiniLM-L12-v2-Q6_K-GGUF	Excluído	XII
8	sizrox/paraphrase-multilingual-mpnet-base-v2-Q8_0-GGUF	Excluído	XII
9	ibm-granite/granite-embedding-278m-multilingual	Incluído	
10	sentence-transformers/distiluse-base-multilingual-cased-v1	Excluído	XII

Fonte: Elaborado pelos Autores (2025).

Como resultado, foram selecionados os seguintes modelos de *embedding* para continuidade da implementação e testes: p*araphrase-multilingual-MiniLM-L12-v2, multilingual-e5-base, gte-multilingual-base, LaBSE, granite-embedding-278m-multilingual, distiluse-base-multilingual-cased-v2, multilingual-e5-small, sentence-embedding-LaBSE* e *static-similarity-mrl-multilingual-v1*.

Pipeline de Implementação

Para a terceira etapa, foi implementado um pipeline de ingestão e consulta de dados utilizando os bancos de dados vetoriais definidos na primeira etapa. Esta etapa envolveu: preparação dos dados, geração de *embeddings* e indexação dos dados nos bancos vetoriais.

[7] Link do Quadro 5 completo: https://doi.org/10.5281/zenodo.17345247

[8] Link do Quadro 6 completo: https://doi.org/10.5281/zenodo.17345247

a) Preparação dos dados

O arquivo da base de dados criada no formato .XLSX, contendo os registros das ferramentas de IA cadastradas e as respectivas avaliações e comentários, foi convertido para o formato JSONL (JSON Lines), formato pré-definido para a indexação e experimento nos bancos vetoriais.

O arquivo JSONL[9] gerado a partir da consolidação dos dados foi estruturado para que cada linha do arquivo represente um objeto JSON independente, contendo dois campos principais: 1) id: identificador único da ferramenta, conforme registrado na base original e 2) texto: campo textual unificado que agrega as informações descritivas da ferramenta, suas avaliações e comentários, organizados no formato "Nome da Coluna: Valor" e separados por quebras de linhas.

Para o experimento proposto neste trabalho, foram utilizados 162 registros completos, cada um representando uma ferramenta de IA e suas respectivas descrições, avaliações e comentários.

b) Geração de *embeddings* e indexação

A fase de vetorização dos dados textuais e a ingestão deles nos bancos vetoriais selecionados consistiu também na utilização de diferentes modelos de *embedding* selecionados para o experimento. Para isso, desenvolveu-se um *script Python*[10] responsável por: carregar e processar o arquivo JSONL, aplicar os modelos de *embeddings* previamente selecionados para gerar a representação vetorial dos textos e ingestão desses vetores em cada um dos bancos de dados vetoriais escolhidos, com persistência local.

Cada um dos modelos de *embedding* foi testado em combinação com cada um dos bancos vetoriais, de maneira a permitir uma avaliação de desempenho dos diferentes pares de configuração.

Avaliação da Relevância e Diversidade

A avaliação da relevância e diversidade na recuperação de informação vetorial proposta neste estudo implementou um *script Python* (av1.py), que operacionaliza a comparação entre os diferentes bancos vetoriais (Chroma, Milvus, Pgvector, Qdrant, Txtai e Weaviate) combinados com os modelos de *embeddings* (*paraphrase-multilingual-MiniLM-L12-v2*, *multilingual-e5-base*, *gte-multilingual-base*, *LaBSE*, *granite-embedding-278m-multilingual*, *distiluse-base-multilingual-cased-v2*, *multilingual-e5-small*, *sentence-embedding-LaBSE* e *static-similarity-mrl-multilingual-v1)*.

Para a execução da avaliação comparativa, o sistema foi testado em um *test suite* de duas consultas pré-definidas: 1) "análise de sentimento" e 2) "correção de texto", como documentos anotados manualmente quanto à relevância e categorização temática. Além disso, foram comparados dois cenários: a) *Baseline,* sendo os resultados obtidos diretamente do banco pela busca vetorial e b) MMR, que reordena os resultados iniciais, variando o parâmetro λ para equilibrar similaridade com a consulta (relevância) e dissimilaridade entre documentos (diversidade).

Com isso, foi possível avaliar tanto a eficiência dos modelos de *embeddings* na recuperação de documentos relevantes quanto a capacidade dos bancos vetoriais em fornecer diversidade informacional, reduzindo redundâncias e ampliando a cobertura temática.

A seguir apresentam-se os resultados empíricos obtidos a partir da execução desse protocolo.

a) Relevância e Ranking

A avaliação da relevância no processo de recuperação semântica foi conduzida a partir da métrica Recall@k, para verificar a capacidade das combinações entre bancos vetoriais e modelos de *embedding* em recuperar documentos pertinentes

[9] Link do arquivo JSONL: https://doi.org/10.5281/zenodo.17345247

[10] Link com todos os scripts: https://doi.org/10.5281/zenodo.17345247

às consultas propostas. As Tabelas 1 e 2 apresentam os resultados para as consultas "*análise de sentimento"* e "*correção de texto*", respectivamente, ordenados a partir da coluna *Baseline*, de modo a evidenciar inicialmente os pares banco vetorial/*embedding* com maior capacidade de recuperação em sua configuração padrão, antes da aplicação do *reranking* via *Maximal Marginal Relevance* (MMR).

Tabela 1: Os 5 primeiros e os 5 últimos resultados da avaliação Recall@k para a consulta "análise de sentimento" [11]

Nº	BD Vetorial	Modelo de *Embedding*	Baseline	MMR (λ=0.7)	MMR (λ=0.5)	MMR (λ=0.3)
1	Chroma	multilingual-e5-base	0,4167	0,3333	0,1667	0,1667
2	Chroma	gte-multilingual-base	0,4167	0,4167	0,2500	0,1667
3	Chroma	LaBSE	0,4167	0,4167	0,2500	0,1667
4	Chroma	granite-embedding-278m-multilingual	0,4167	0,3333	0,1667	0,1667
5	Chroma	distiluse-base-multilingual-cased-v2	0,4167	0,3333	0,2500	0,1667
...	...	...	...	...	...	...
50	txtai	paraphrase-multilingual-MiniLM-L12-v2	0,3333	0,4167	0,3333	0,1667
51	txtai	multilingual-e5-small	0,3333	0,3333	0,1667	0,0833
52	Weaviate	distiluse-base-multilingual-cased-v2	0,3333	0,3333	0,2500	0,0833
53	Weaviate	multilingual-e5-small	0,3333	0,3333	0,1667	0,0833
54	txtai	distiluse-base-multilingual-cased-v2	0,1667	0,3333	0,2500	0,1667

Fonte: Elaborado pelos Autores (2025).

No caso da consulta "*análise de sentimento*" (Tabela 1), observou-se que praticamente todas as combinações entre bancos vetoriais e modelos de *embedding* atingiram Recall@k igual a 0,4167 na configuração *baseline*, com destaque inicial para o Chroma e o Milvus, associados a *embeddings* como multiligual-e5-base, gte-multilingual-base e LaBSE. Essa uniformidade na linha de base indica que, nesse cenário específico, a distinção entre soluções está menos ligada ao banco vetorial e mais à estratégia de *reranking*. Observa-se, contudo, que a aplicação do MMR, em especial com valores mais baixos de λ (0,5 e 0,3), reduziu significativamente o *recall*, sinalizando uma perda de documentos relevantes em favor da diversidade. Assim, para tarefas em que a exaustividade da recuperação é prioritária, o uso direto da configuração *baseline* se mostra mais vantajoso.

Tabela 2: Os 5 primeiros e os 5 últimos resultados da avaliação Recall@k para a consulta "correção de texto" [12]

Nº	BD Vetorial	Modelo de *Embedding*	Baseline	MMR (λ=0.7)	MMR (λ=0.5)	MMR (λ=0.3)
1	Chroma	paraphrase-multilingual-MiniLM-L12-v2	0,4167	0,4167	0,2500	0,1667
2	Chroma	gte-multilingual-base	0,4167	0,2500	0,0833	0,0833
3	Chroma	granite-embedding-278m-multilingual	0,4167	0,2500	0,1667	0,0833
4	Chroma	distiluse-base-multilingual-cased-v2	0,4167	0,2500	0,0833	0,0833
5	Weaviate	paraphrase-multilingual-MiniLM-L12-v2	0,4167	0,4167	0,2500	0,1667
...	...	...	...	...	...	...
50	txtai	LaBSE	0,3333	0,2500	0,0833	0,0833
51	txtai	distiluse-base-multilingual-cased-v2	0,3333	0,2500	0,2500	0,1667
52	txtai	multilingual-e5-small	0,3333	0,2500	0,2500	0,2500
53	txtai	sentence-embedding-LaBSE	0,3333	0,2500	0,0833	0,0833
54	txtai	static-similarity-mrl-multilingual-v1	0,3333	0,2500	0,0833	0,1667

Fonte: Elaborado pelos Autores (2025).

Para a consulta "correção de texto" (Tabela 2), o cenário é ligeiramente distinto; embora também haja uniformidade inicial (Recall@k =0,4167 em muitas combinações), o modelo paraphrase-multilingual-MiniLM-L12-v2 se destacou de forma

[11] Link da Tabela 1 completa: https://doi.org/10.5281/zenodo.17345247

[12] Link da Tabela 2 completa: https://doi.org/10.5281/zenodo.17345247

consistente, obtendo bons resultados em diferentes bancos vetoriais (Chroma, Weaviate e Milvus). Da mesma forma que na primeira consulta, o uso do MMR implicou queda no *recall*, assinalando o dilema clássico entre relevância e diversidade já apontado na literatura de recuperação de informação (Carpineto; Romano, 2012; Radlinski; Craswell, 2017).

Em complemento à análise de recall, as Tabelas 3 e 4 apresentam os resultados referentes ao *ranking* dos documentos recuperados, mensurados por meio da métrica nDCG@k igualmente ordenados pela coluna *Baseline*. Essa métrica permite avaliar não só a presença de documentos relevantes, mas também a posição em que eles são apresentados ao usuário, o que é decisivo em sistema de recomendação e curadoria de informação.

Na consulta "análise de sentimento" (Tabela 3) observou-se que muitas combinações atingiram desempenho máximo (Baseline = 1,0000), notadamente aquelas envolvendo Chroma, Weaviate, Milvus, Qdrant e pgvector, associados à modelos como multilingual-e5-base, gte-multilingual-base e LaBSE. Esse resultado evidencia a consistência do posicionamento dos itens mais relevantes quando não há intervenção de *reranking*. No entanto, com a aplicação do MMR, especialmente para $\lambda = 0,5$ e $\lambda = 0,3$, houve queda acentuada nos valores de ordenação ótima dos documentos. Ainda assim, alguns *embeddings*, como paraphrase-multilingual-MiniLM-L12-v2, mantiveram bons resultados após o *reranking* sugerindo maior resiliência a esse tipo de ajuste.

Tabela 3: Os 5 primeiros e os 5 últimos resultados da avaliação nDCG@k para a consulta "análise de sentimento" [13]

Nº	BD Vetorial	Modelo de *Embedding*	Baseline	MMR (λ=0.7)	MMR (λ=0.5)	MMR (λ=0.3)
1	Chroma	multilingual-e5-base	1,0000	0,8539	0,5531	0,4852
2	Chroma	gte-multilingual-base	1,0000	1,0000	0,6844	0,5087
3	Chroma	LaBSE	1,0000	1,0000	0,7227	0,4704
4	Chroma	granite-embedding-278m-multilingual	1,0000	0,8688	0,5531	0,5087
5	Chroma	sentence-embedding-LaBSE	1,0000	1,0000	0,7227	0,4704
...	...	...	...	...	...	...
50	pgvector	distiluse-base-multilingual-cased-v2	0,8688	0,8688	0,6844	0,3392
51	pgvector	multilingual-e5-small	0,8688	0,8304	0,5087	0,3392
52	txtai	multilingual-e5-small	0,8688	0,8304	0,5087	0,3392
53	txtai	paraphrase-multilingual-MiniLM-L12-v2	0,8539	0,9563	0,7860	0,4704
54	txtai	distiluse-base-multilingual-cased-v2	0,5531	0,8688	0,6992	0,4365

Fonte: Elaborado pelos Autores (2025).

Para a consulta "correção de texto" (Tabela 4), o comportamento foi mais heterogêneo. Embora diferentes pares de banco vetorial/*embedding* tenham atingido o *Baseline* de 0,4167, o destaque novamente recai sobre o modelo paraphrase-multilingual-MiniLM-L12-v2, que apresentou desempenho consistente em diferentes bases (Chroma, Weaviate, Milvus, Qdrant e pgvector). Em contrapartida, *embeddings* como multilingual-e5-small obtiveram valores mais baixos de *Baseline* (0,3333), mas mostraram alguma recuperação em cenários de *reranking*, indicando potencial em contextos nos quais a diversidade seja igualmente valorizada.

[13] Link da Tabela 3 completa: https://doi.org/10.5281/zenodo.17345247

Tabela 4: Os 5 primeiros e os 5 últimos resultados da avaliação nDCG@k para a consulta "correção de texto"[14]

Nº	BD Vetorial	Modelo de *Embedding*	Baseline	MMR (λ=0.7)	MMR (λ=0.5)	MMR (λ=0.3)
1	Chroma	paraphrase-multilingual-MiniLM-L12-v2	1,0000	1,0000	0,6164	0,4365
2	Chroma	multilingual-e5-base	0,8539	0,5087	0,3392	0,3392
3	Chroma	gte-multilingual-base	1,0000	0,6399	0,3392	0,3392
4	Chroma	LaBSE	0,8304	0,5087	0,4852	0,3392
5	Chroma	granite-embedding-278m-multilingual	1,0000	0,6844	0,4704	0,3392
...	...	...	...	...	...	...
50	txtai	granite-embedding-278m-multilingual	1,0000	0,6844	0,4704	0,3392
51	txtai	distiluse-base-multilingual-cased-v2	0,8304	0,6844	0,6164	0,4365
52	txtai	multilingual-e5-small	0,8688	0,6992	0,6399	0,6399
53	txtai	sentence-embedding-LaBSE	0,8304	0,6399	0,3392	0,3392
54	txtai	static-similarity-mrl-multilingual-v1	0,7860	0,6164	0,3392	0,4266

Fonte: Elaborado pelos Autores (2025).

Com isso, observa-se que os resultados obtidos nas duas consultas analisadas ("análise de sentimento" e "correção de texto"), o desempenho das combinações entre bancos vetoriais e modelos de *embeddings* foi relativamente uniforme na configuração *baseline*, com destaque para o Chroma e os modelos LaBSE, gte-multilingual-base, multilingual-e5-base. A aplicação do reranking via MMR, embora tenha promovido maior diversidade nos resultados, reduziu o valor do Recall@k e do nDCG@k em praticamente todos os casos.

b) Diversidade e Equilíbrio

A avaliação da diversidade e do equilíbrio buscou analisar em que medida os diferentes bancos vetoriais e modelos de *embedding* puderam recuperar não apenas documentos relevantes, mas também representações que contemplam variação semântica (diversidade) e equilíbrio entre tópicos (α-nDCG@k). As Tabelas 5 e 6 apresentam os resultados de diversidade (ILD@k), enquanto as Tabelas 7 e 8 sintetizam o equilíbrio entre relevância e diversidade, todas ordenadas a partir da coluna *Baseline*.

No caso da consulta "análise de sentimento" (Tabela 5), os melhores desempenhos de diversidade concentraram-se em combinações envolvendo o modelo LaBSE (e sua variação sentence-embedding-LaBSE), independentemente do banco vetorial adotado. Esses pares apresentaram ILD@k de 0,9819 na configuração Baseline, com manutenção de valores aproximados mesmo após a aplicação do MMR.

Em segundo plano, observou-se o modelo multilingual-e5-small, que, embora com valores ligeiramente inferiores (≈ 0,9704), manteve-se consistente em diferentes bases. Por outro lado, *embeddings* como paraphrase-multilingual-MiniLM-L12-v2 exibiram valores mais modestos (≈ 0,8466), sugerindo que seu foco em precisão linguística não se traduz em ampla cobertura semântica.

[14] Link da Tabela 4 completa: https://doi.org/10.5281/zenodo.17345247

Tabela 5: Os 5 primeiros e os 5 últimos resultados da avaliação ILD@k para a consulta "análise de sentimento" [15]

Nº	BD Vetorial	Modelo de *Embedding*	Baseline	MMR (λ=0.7)	MMR (λ=0.5)	MMR (λ=0.3)
1	Chroma	LaBSE	0,9819	0,9835	0,9017	0,7963
2	Chroma	sentence-embedding-LaBSE	0,9819	0,9835	0,9017	0,7963
3	Weaviate	LaBSE	0,9819	0,9835	0,9017	0,7963
4	Weaviate	sentence-embedding-LaBSE	0,9819	0,9835	0,9017	0,7963
5	Milvus	LaBSE	0,9819	0,9835	0,9017	0,7963
...	...	...	...	...	...	...
50	txtai	distiluse-base-multilingual-cased-v2	0,8293	0,9442	0,8859	0,7632
51	Weaviate	paraphrase-multilingual-MiniLM-L12-v2	0,7786	0,7451	0,7612	0,7846
52	Qdrant	paraphrase-multilingual-MiniLM-L12-v2	0,7786	0,7451	0,7612	0,7846
53	pgvector	paraphrase-multilingual-MiniLM-L12-v2	0,7786	0,7451	0,7612	0,7846
54	txtai	paraphrase-multilingual-MiniLM-L12-v2	0,7324	0,7873	0,7446	0,7048

Fonte: Elaborado pelos Autores (2025).

Para a consulta "correção de texto" (Tabela 6), destacou-se o *embedding* multilingual-e5-base, que obteve desempenho de 0,9615 em todas as bases vetoriais, seguido pelo modelo multilingual-e5-small com 0,9584. Tais resultados indicam que esses modelos são particularmente eficazes em capturar diversidade semântica em tarefas voltadas à qualidade textual. Os demais *embeddings*, como LaBSE e gte-multilingual-base, também alcançaram bom desempenho, 0,9175 e 0,9227 respectivamente. Assim como observado na consulta anterior, modelos voltados à paráfrase, como MiniLM-L12-v2, apresentaram menor capacidade de diversificação.

Tabela 6: Os 5 primeiros e os 5 últimos resultados da avaliação ILD@k para a consulta "correção de texto"[16]

Nº	BD Vetorial	Modelo de *Embedding*	Baseline	MMR (λ=0.7)	MMR (λ=0.5)	MMR (λ=0.3)
1	Chroma	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
2	Weaviate	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
3	Milvus	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
4	Qdrant	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
5	pgvector	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
...	...	...	...	...	...	...
50	Qdrant	paraphrase-multilingual-MiniLM-L12-v2	0,8719	0,8732	0,7836	0,7116
51	pgvector	paraphrase-multilingual-MiniLM-L12-v2	0,8719	0,8732	0,7836	0,7116
52	Chroma	paraphrase-multilingual-MiniLM-L12-v2	0,8531	0,8732	0,7836	0,7116
53	Milvus	paraphrase-multilingual-MiniLM-L12-v2	0,8531	0,8732	0,7836	0,7116
54	txtai	distiluse-base-multilingual-cased-v2	0,8249	0,8184	0,8069	0,7702

Fonte: Elaborado pelos Autores (2025).

No que se refere ao equilíbrio entre relevância e diversidade, os resultados para a consulta "análise de sentimento" (Tabela 7) mostraram maior dispersão. O destaque foi o *embedding* paraphrase-multilingual-MiniLM-L12-v2, especialmente na base txtai, que atingiu valores de α-nDCG@k de 0,3528 no *Baseline* e desempenho ainda melhor com λ = 0,3 (0,5523). Já os modelos distiluse-base-multiligual-cased-v2 e granite-embedding-278m-multilingual apresentaram equilíbrio intermediário, enquanto LaBSE e multilingual-e5-base ocuparam posições inferiores, sugerindo menor capacidade de balancear diversidade sem sacrificar relevância.

[15] Link da Tabela 5 completa: https://doi.org/10.5281/zenodo.17345247

[16] Link da Tabela 6 completa: https://doi.org/10.5281/zenodo.17345247

Tabela 7: Os 5 primeiros e os 5 últimos resultados da avaliação α-nDCG@k para a consulta "análise de sentimento" [17]

Nº	BD Vetorial	Modelo de *Embedding*	Baseline	MMR (λ=0.7)	MMR (λ=0.5)	MMR (λ=0.3)
1	txtai	paraphrase-multilingual-MiniLM-L12-v2	0,3528	0,2391	0,3528	0,5523
2	txtai	distiluse-base-multilingual-cased-v2	0,3501	0,2847	0,3345	0,4341
3	Chroma	distiluse-base-multilingual-cased-v2	0,2669	0,2847	0,3345	0,4677
4	Milvus	distiluse-base-multilingual-cased-v2	0,2669	0,2847	0,3345	0,4677
5	Weaviate	distiluse-base-multilingual-cased-v2	0,2532	0,3137	0,3545	0,4410
...	...	...	...	...	...	...
50	Weaviate	multilingual-e5-small	0,0752	0,0752	0,1031	0,1068
51	Milvus	multilingual-e5-small	0,0752	0,0752	0,1031	0,1068
52	Qdrant	multilingual-e5-small	0,0752	0,0752	0,1031	0,1068
53	pgvector	multilingual-e5-small	0,0752	0,0752	0,1031	0,1068
54	txtai	multilingual-e5-small	0,0752	0,0752	0,1031	0,1068

Fonte: Elaborado pelos Autores (2025).

Para a consulta "correção de texto" (Tabela 8), a configuração Baseline evidenciou novamente uma superioridade dos modelos multilingual-e5-base e multilingual-e5-small, ambos com valores de α-nDCG@k superiores a 0,95 em diferentes bancos vetoriais. Em contrapartida, modelos como distulise-base-multilingual-cased-v2 e paraphrase-multilingual-MiniLM-L12-v2 obtiveram desempenho inferior (≈0,82-0,87), evidenciando a dificuldade de manter equilíbrio quando se privilegia ora diversidade, ora a precisão linguística.

Tabela 8: Os 5 primeiros e os 5 últimos resultados da avaliação α-nDCG@k para a consulta "correção de texto" [18]

Nº	BD Vetorial	Modelo de *Embedding*	Baseline	MMR (λ=0.7)	MMR (λ=0.5)	MMR (λ=0.3)
1	Chroma	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
2	Weaviate	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
3	Milvus	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
4	Qdrant	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
5	pgvector	multilingual-e5-base	0,9615	0,8998	0,8724	0,8724
...	...	...	...	...	...	...
50	Qdrant	paraphrase-multilingual-MiniLM-L12-v2	0,8719	0,8732	0,7836	0,7116
51	pgvector	paraphrase-multilingual-MiniLM-L12-v2	0,8719	0,8732	0,7836	0,7116
52	Chroma	paraphrase-multilingual-MiniLM-L12-v2	0,8531	0,8732	0,7836	0,7116
53	Milvus	paraphrase-multilingual-MiniLM-L12-v2	0,8531	0,8732	0,7836	0,7116
54	txtai	distiluse-base-multilingual-cased-v2	0,8249	0,8184	0,8069	0,7702

Fonte: Elaborado pelos Autores (2025).

A partir dos resultados apresentados, pode-se observar que a análise da diversidade e equilíbrio evidenciou desempenhos contratantes entre os modelos de *embedding* avaliados. O LaBSE e suas variações destacaram-se pela alta diversidade semântica (ILD@k), demonstrando maior capacidade de capturar variações de significado entre os documentos recuperados, independentemente do banco vetorial utilizado. Em contrapartida, modelos mais compactos ou voltados à paráfrase – como o *paraphrase-multilingual-MiniLM-L12-v2* –apresentaram menor amplitude semântica, ainda que tenham se sobressaído em equilíbrio (α-nDCG@k, sobretudo em cenários de ajuste de MMR (λ = 0,3). Já os modelos da família multilingual-e5(*base* e *small*) mostraram desempenho mais estável e balanceado nas duas consultas ("análise de sentimento" e "correção de texto"), alcançando bons níveis tanto de diversidade quanto de equilíbrio, em particular, em tarefas relacionadas à qualidade textual.

[17] Link da Tabela 7 completa: https://doi.org/10.5281/zenodo.17345247

[18] Link da Tabela 8 completa: https://doi.org/10.5281/zenodo.17345247

Avaliação Sistemática de Eficiência Computacional

A análise de eficiência computacional teve como objetivo examinar o desempenho dos diferentes bancos vetoriais e modelos de *embeddings* sob a perspectiva de tempo de resposta, consumo de recursos e capacidade de escalabilidade. Para isso, foram utilizadas as métricas de latência média, latência p95, uso médio de CPU, uso médio de RAM e Queries per Second (QPS). Todas as tabelas desta subseção foram ordenadas a partir da coluna "Produto Escalar", de modo a destacar inicialmente as combinações com melhor desempenho nessa dimensão de similaridade.

a) Tempo de resposta

Os resultados de latência média (Tabela 9) evidenciaram que soluções leves, como Weaviate, txtai e pgvector, alcançaram os menores tempos de resposta, situando-se na faixa de 2,2 a 3,5 ms em suas melhores combinações. Destacam-se os pares Weaviate + granite-embedding-278m-multilingual (2,22 ms) e txtai + gte-multilingual-base (2,50 ms). Já bancos mais robustos, como Milvus e Qdrant, apresentaram latência média mais alta, variando entre 3,8 e 17 ms no caso do Qdrant, e chegando a 46 ms com o pgvector em determinadas configurações.

Tabela 9: Os 5 primeiros e os 5 últimos resultados da avaliação Latência Média (ms) [19]

Nº	BD Vetorial	Modelo de *Embedding*	Produto Escalar	Cosseno	Euclidiana (L2)
1	Weaviate	granite-embedding-278m-multilingual	2,22	2,68	2,93
2	txtai	gte-multilingual-base	2,50	2,94	2,60
3	pgvector	multilingual-e5-small	2,50	2,59	2,71
4	Weaviate	distiluse-base-multilingual-cased-v3	2,56	2,34	2,82
5	pgvector	paraphrase-multilingual-MiniLM-L12-v6	2,61	2,50	2,97
...	...	...	...	...	...
50	pgvector	distiluse-base-multilingual-cased-v6	45,67	45,78	45,65
51	pgvector	granite-embedding-278m-multilingual	45,89	45,97	46,58
52	pgvector	gte-multilingual-base	46,25	45,91	46,36
53	pgvector	multilingual-e5-base	46,64	46,39	46,23
54	pgvector	static-similarity-mrl-multilingual-v5	46,66	46,42	46,61

Fonte: Elaborado pelos Autores (2025).

A análise da latência p95 (Tabela 10), que considera os piores casos, reforça esse padrão. Novamente, txtai e pgvector obtiveram desempenhos melhores (≈3 a 4 ms), enquanto Qdrant e pgvector com *embeddings* LaBSE e similares alcançaram tempos superiores a 40 ms, indicando maior variabilidade em cenários de maior carga.

Tabela 10: Os 5 primeiros e os 5 últimos resultados da avaliação Latência p95 (ms) [20]

Nº	BD Vetorial	Modelo de *Embedding*	Produto Escalar	Cosseno	Euclidiana (L2)
1	txtai	gte-multilingual-base	2,78	3,22	2,95
2	Weaviate	granite-embedding-278m-multilingual	3,20	4,91	5,46
3	txtai	paraphrase-multilingual-MiniLM-L12-v7	3,40	3,61	3,26
4	pgvector	multilingual-e5-small	3,47	3,87	4,04
5	pgvector	paraphrase-multilingual-MiniLM-L12-v6	3,71	3,44	3,51
...	...	...	...	...	...
50	pgvector	multilingual-e5-base	48,64	49,66	49,52
51	pgvector	gte-multilingual-base	48,81	48,05	48,74
52	pgvector	static-similarity-mrl-multilingual-v5	48,85	49,11	49,82
53	pgvector	distiluse-base-multilingual-cased-v6	48,99	48,63	48,26
54	pgvector	sentence-embedding-LaBSE	49,11	48,52	49,57

Fonte: Elaborado pelos Autores (2025).

[19] Link da Tabela 9 completa: https://doi.org/10.5281/zenodo.17345247

[20] Link da Tabela 10 completa: https://doi.org/10.5281/zenodo.17345247

Assim, os resultados sugerem que, para aplicações sensíveis ao tempo de resposta, como chatbots e sistemas interativos em tempo real, bancos como Weaviate e txtai se mostram mais adequados. Já Milvus e Qdrant podem compensar em cenários que demandam maior robustez de indexação.

b) Uso de Recursos Computacionais

A Tabela 11, que se refere ao uso de CPU, mostrou que o Weaviate foi o mais eficiente, com consumo médio inferior a 10% em várias combinações de *embeddings* (como granite-embedding e LaBSE). Em contrapartida, soluções como txtai atingiram picos acima de 600%, refletindo elevado custo de processamento, possivelmente relacionado ao seu design de execução em memória. Milvus e Qdrant mantiveram consumos intermediários, entre 20% e 40%.

Tabela 11: Os 5 primeiros e os 5 últimos resultados da avaliação de Uso de Recursos – Média CPU (%) [21]

Nº	BD Vetorial	Modelo de *Embedding*	Produto Escalar	Cosseno	Euclidiana (L2)
1	Weaviate	granite-embedding-278m-multilingual	4,54	4,54	22,69
2	Weaviate	sentence-embedding-LaBSE	4,56	7,53	21,22
3	Weaviate	multilingual-e5-small	5,94	3,05	24,58
4	Weaviate	paraphrase-multilingual-MiniLM-L12-v3	7,56	6,01	18,12
5	Weaviate	distiluse-base-multilingual-cased-v3	7,58	5,97	14,98
...	...	...	...	...	...
50	txtai	distiluse-base-multilingual-cased-v7	461,31	459,39	504,89
51	txtai	gte-multilingual-base	485,00	408,55	341,77
52	txtai	paraphrase-multilingual-MiniLM-L12-v7	517,62	614,34	661,51
53	txtai	granite-embedding-278m-multilingual	603,31	414,66	389,43
54	txtai	multilingual-e5-base	611,04	515,82	540,03

Fonte: Elaborado pelos Autores (2025).

No que se refere ao uso de memória RAM (Tabela 12), o pgvector demonstrou-se a opção mais econômica, mantendo consumo entre 176 e 178 MB em todas as combinações. Milvus apresentou valores na faixa de 186 e 195 MB, seguido pelo Weaviate com cerca de 2020 e 228 MB. Já Qdrant e Chroma apresentaram um consumo maior, com valores entre 275 MB (Qdrant) e mais de 370 MB (Chroma). O destaque negativo ficou com o txtai, que apresentou uso de RAM na ordem de 2 a 4 GB, chegando a 4,4 GB em algumas configurações, o que compromete sua viabilidade em ambientes de recursos limitados.

Tabela 12: Os 5 primeiros e os 5 últimos resultados da avaliação Uso de Recursos – Média RAM (MB) [22]

Nº	BD Vetorial	Modelo de *Embedding*	Produto Escalar	Cosseno	Euclidiana (L2)
1	pgvector	multilingual-e5-small	176,26	173,31	176,14
2	pgvector	paraphrase-multilingual-MiniLM-L12-v6	176,42	173,48	176,31
3	pgvector	distiluse-base-multilingual-cased-v6	176,69	173,69	176,55
4	pgvector	LaBSE	176,78	173,90	176,68
5	pgvector	multilingual-e5-base	177,14	173,89	176,78
...	...	...	...	...	...
50	txtai	LaBSE	3248,91	3245,59	3248,64
51	txtai	sentence-embedding-LaBSE	3250,67	3247,53	3250,29
52	txtai	multilingual-e5-base	3327,55	3323,85	3326,54
53	txtai	granite-embedding-278m-multilingual	3838,77	3834,76	3837,89
54	txtai	gte-multilingual-base	4411,17	4406,85	4409,75

Fonte: Elaborado pelos Autores (2025).

[21] Link da Tabela 11 completa: https://doi.org/10.5281/zenodo.17345247

[22] Link da Tabela 12 completa: https://doi.org/10.5281/zenodo.17345247

Esses resultados indicam que, para aplicações em infraestrutura restrita, pgvector e Milvus são alternativas mais adequadas, enquanto Chroma e txtai exigem ambientes com maior disponibilidade de memória.

c) Capacidade de Escalabilidade

A métrica de *Queries per Second* (QPS), apresentada na Tabela 13 e igualmente ordenada pela coluna "Produto Escalar", evidencia a capacidade dos bancos em lidar com múltiplas requisições simultâneas. O destaque foi o Weaviate, que atingiu valores superiores a 450 QPS em combinação com granite-embedding-278m-multilingual. Pgvector e txtai também apresentaram bons resultados, na faixa de 370 a 400 QPS. Chroma e Milvus mantiveram desempenho intermediário, variando entre 250 e 300 QPS, enquanto o Qdrant apresentou a menor escalabilidade, com valores médios próximos a 60QPS, em todas as combinações.

Tabela 13: Os 5 primeiros e os 5 últimos resultados da avaliação de consultas processadas por segundo (QPS -Estimado) [23]

Nº	BD Vetorial	Modelo de *Embedding*	Produto Escalar	Cosseno	Euclidiana (L2)
1	Weaviate	granite-embedding-278m-multilingual	451,32	373,54	341,66
2	pgvector	multilingual-e5-small	400,21	386,74	369,62
3	txtai	gte-multilingual-base	399,24	340,15	384,63
4	Weaviate	distiluse-base-multilingual-cased-v3	390,15	427,20	355,20
5	pgvector	paraphrase-multilingual-MiniLM-L12-v6	382,92	399,98	337,06
...	...	...	...	...	...
50	pgvector	distiluse-base-multilingual-cased-v6	21,89	21,84	21,90
51	pgvector	granite-embedding-278m-multilingual	21,79	21,75	21,47
52	pgvector	gte-multilingual-base	21,62	21,78	21,57
53	pgvector	multilingual-e5-base	21,44	21,56	21,63
54	pgvector	static-similarity-mrl-multilingual-v5	21,43	21,54	21,45

Fonte: Elaborado pelos Autores (2025).

Esse cenário sugere que Weaviate é a solução mais indicada para aplicações de grande escala, em que a velocidade de atendimento a múltiplos usuários é fator crítico. Já Qdrant, embora eficiente em termos de robustez de indexação, pode tornar um gargalo em sistemas de alta demanda.

4. Discussão

Os resultados obtidos evidenciam o potencial dos bancos de dados vetoriais aliados a modelos de *embedding* multilíngues para a recuperação semântica em língua portuguesa. A análise comparativa demostrou que determinadas combinações apresentaram maior equilíbrio entre relevância, diversidade e desempenho computacional, confirmando a importância de avaliar não apenas a precisão das consultas, mas também aspectos relacionados à escalabilidade e ao custo de processamento.

Verificou-se que soluções dedicadas, como Milvus, Qdrant e Weaviate, tendem a apresentar melhor desempenho em métricas de latência e *queries per second,* enquanto alternativas mais leves, como Chroma e pgvector, mostraram-se adequadas em cenários com menor volume de dados ou recursos computacionais restritos. No que se refere aos modelos de *embedding*, as arquiteturas multilíngues disponibilizadas no *Hugging Face* demonstraram capacidade de generalização satisfatória para o português, ainda que o desempenho varie conforme o tipo de tarefa avaliada.

As avaliações de relevância e diversidade indicaram que o uso de estratégias como o M*aximal Marginal Relevance (*MMR) contribui para reduzir redundâncias e ampliar a cobertura temática dos resultados, aspecto fundamental em sistemas de

[23] Link da Tabela 13 completa: https://doi.org/10.5281/zenodo.17345247

recomendação e em contextos nos quais a diversidade informacional assume papel estratégico. À vista disso, a avaliação dos resultados de relevância e ranking aponta para os seguintes achados principais: primeiramente, combinações que envolvam *embeddings* multilíngues (LaBSE, gte-multilingual-base, multilingual-e5-base) apresentaram robustez; além disso, combinações envolvendo Chroma e modelos multilíngues como LaBSE e gte-multilingual-base apresentaram melhor desempenho; por outro lado, o modelo paraphrase-multilingual-MiniLM-L12-v2 apresenta superioridade em consultas que demandam maior precisão linguística à língua portuguesa; por fim, observou-se que a introdução do MMR, embora traga ganhos em diversidade, exige calibração cuidadosa do parâmetro λ para não comprometer a recuperação de documentos relevantes.

Quanto à avaliação dos resultados de diversidade e equilíbrio trouxeram as seguintes conclusões: de um lado, modelos como LaBSE e multilingual-e5-small mostraram-se mais eficazes em termos de diversidade (ILD@k), garantindo ampla cobertura semântica; de outro, *embeddings* da família multilingual-e5 demonstraram ser particularmente melhores em consultas ligadas a tarefas linguísticas (“correção de texto”), equilibrando diversidade e relevância em diferentes cenários. Adicionalmente, o modelo paraphrase-multilingual-MiniLM-L12-v2, embora com menor diversidade, destacou-se em equilíbrio (α-nDCG@k) em contextos nos quais a variação semântica é menos crítica. Dessa forma, a escolha entre maximizar diversidade ou buscar equilíbrio depende diretamente do objetivo da recuperação, indicando a necessidade de calibragem criteriosa do parâmetro λ no MMR.

Portanto, os resultados demonstram que a diversidade e equilíbrio constituem dimensões complementares à relevância, sendo necessários para aplicações onde a pluralidade de perspectivas e a cobertura de subtemas desempenham papel central.

A avaliação de desempenho reforçou que a escolha do banco de dados vetorial deve considerar também a disponibilidade de recursos de hardware, dado o impacto direto no tempo de resposta e no consumo de CPU e RAM. Os resultados da análise de eficiência computacional evidenciam que o desempenho dos bancos vetoriais e modelos de *embeddings* altera significativamente conforme o tipo de métrica e o contexto de uso. Em termos de reposta, soluções mais leves, como Weaviate, txtai e pgvector, apresentaram latências médias e p95 inferiores. Por outro lado, outros bancos, como Milvus e Qdrant, embora apresentem tempos de resposta mais elevados, oferecem maior estabilidade em cenários de alta carga e complexidade de indexação.

No que se refere ao uso de recursos computacionais, o Weaviate demonstrou-se o mais eficiente no consumo de CPU, enquanto o pgvector obteve melhor desempenho quanto à economia de memória RAM, mantendo baixo o consumo mesmo em combinações com *embeddings* de maior porte. Porém, o txtai apresentou picos elevados de uso de CPU e RAM, o que restringe sua adoção em infraestruturas limitadas. Já quanto à capacidade de escalabilidade, mostrou-se que o Weaviate pode processar mais de 450 requisições por segundo (QPS), seguido por pgvector e txtai.

Em tempo, embora relevantes, os resultados devem ser interpretados considerando algumas limitações. As análises foram realizadas com conjuntos de dados de dimensão reduzida — na ordem de alguns milhares de registros — e com modelos de *embedding* previamente selecionados, o que pode restringir a generalização para domínios distintos ou para cenários de maior escala. Ademais, a avaliação foi delimitada a métricas clássicas de recuperação de informação, não incorporando aspectos subjetivos relacionados à experiência do usuário.

5. Considerações Finais

Este trabalho teve como propósito comparar o desempenho de diferentes combinações entre bancos de dados vetoriais e modelos de *embedding* multilíngues aplicadas à recuperação semântica em língua portuguesa. A partir da questão de

pesquisa proposta — sobre o impacto dessa combinação nas tarefas de busca por similaridade —, foi possível observar variações significativas entre as tecnologias avaliadas.

Os experimentos indicaram que Milvus e Weaviate apresentaram melhor desempenho em situações de maior carga computacional, com tempos de resposta estáveis e boa escalabilidade. O pgvector mostrou-se mais econômico em memória e tempo de execução em ambientes de menor porte. Chroma e pgvector também se revelaram alternativas adequadas para aplicações locais, nas quais simplicidade e eficiência são fatores relevantes.

Entre os modelos de *embedding*, os resultados confirmaram a robustez dos modelos multilíngues disponíveis na plataforma Hugging Face, especialmente LaBSE, multilingual-e5-base e gte-multilingual-base, que mantiveram bom equilíbrio entre relevância e diversidade nas métricas analisadas. O uso do algoritmo Maximal Marginal Relevance (MMR) contribuiu para diversificar os resultados, ainda que com redução pontual no recall, comportamento esperado em ajustes desse tipo.

De modo geral, as evidências mostram que a escolha da combinação banco vetorial–modelo de embedding depende do equilíbrio entre desempenho computacional, qualidade da recuperação e contexto de aplicação. Em projetos que exigem escalabilidade, bancos dedicados como Milvus e Weaviate se mostram mais eficientes; em cenários menores, soluções como pgvector e Chroma oferecem melhor relação entre desempenho e simplicidade.

Esta pesquisa apresenta algumas limitações, como o uso de uma base restrita ao domínio de ferramentas de inteligência artificial, a realização dos testes em ambiente local e o número reduzido de consultas. Pesquisas futuras poderão ampliar o conjunto de dados, incorporar novas métricas de avaliação e explorar ambientes distribuídos, possibilitando análises complementares de custo e desempenho.

Em conjunto, os resultados contribuem para o avanço das avaliações comparativas em língua portuguesa, oferecendo referências empíricas e orientações metodológicas úteis à seleção de combinações adequadas entre bancos vetoriais e modelos de *embedding* em projetos de curadoria digital, observatórios de dados e sistemas de recomendação.

Agradecimentos

Os autores agradecem o apoio acadêmico dos Programas de Pós-Graduação em Gestão da Informação (PPGGI) e em Ciência de Dados (PPGCD) da Universidade Federal do Paraná (UFPR), bem como o apoio institucional do Instituto de Ciência e Tecnologia Itaú (ICTi) e da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), que contribuíram para a realização deste estudo.

Todas as conclusões expressas pelos autores não refletem as opiniões do Itaú Unibanco e do Instituto de Ciência e Tecnologia Itaú (ICTi). Além disso, este material não deve resultar em nenhum processo de natureza comercial. Por fim, todos os dados utilizados neste estudo estão em conformidade com a Lei Geral de Proteção de Dados (LGPD) do Brasil.

Referências

Carbonell, J., & Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 335–336). New York, NY: ACM. https://dl.acm.org/doi/10.1145/290941.291025

Carpineto, C., & Romano, G. (2012). A survey of diversity methods in information retrieval. *ACM Computing Surveys (CSUR), 44*(1), 1–50. https://doi.org/10.1145/2071389.2071390

Carvalho, P., Oliveira, R., Silva, M., & Pereira, T. (2025). Evaluating text representations for unsupervised legal semantic textual similarity in Brazilian Portuguese. *Information and Data Technologies*. Cham: Springer. https://doi.org/10.1007/s44248-025-00052-4

Clarke, C. L. A., Kolla, M., Cormack, G. V., Vechtomova, O., Ashkan, A., Büttcher, S., & MacKinnon, I. (2008). Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 659–666). New York, NY: ACM. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=15004aadabd967ac722a28a9c3bb39cf5bc32605

Fernandes, L. C., Ribeiro, L. S., Castro, M. V. B., Pacheco, L. A. S., & Sandes, E. F. O. (2025). JurisTCU: A Brazilian Portuguese information retrieval dataset with query relevance judgments. *arXiv preprint*. https://arxiv.org/abs/2503.08379

Hartmann, N. S., Fonseca, E. R., Shulby, C., Silva, J., & Aluísio, S. M. (2017). Portuguese word embeddings: Evaluating on word analogies and natural language tasks. *arXiv preprint*. https://arxiv.org/abs/1708.06025

Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems, 20*(4), 422–446. https://faculty.cc.gatech.edu/~zha/CS8803WST/dcg.pdf

Joshi, S. (2025). Introduction to vector databases for generative AI: Applications, performance, future projections, and cost considerations. *International Advanced Research Journal in Science, Engineering and Technology, 12*(2), 79–91. https://doi.org/10.17148/IARJSET.2025.12210

Kerlinger, F. N. (1980). *Metodologia da pesquisa em ciências sociais: Um tratamento conceitual* (H. M. Rotundo, Trad.). São Paulo: EPU.

Latimer, C. (2024). The ultimate guide to vector database success in AI. *Vectorize*. https://vectorize.io/what-is-a-vector-database/
Lewis, P., Perez, E., Pothast, M., Kuznetsov, I., Levy, O., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems, 33*, 9459–9474. Vancouver: Curran Associates. https://dl.acm.org/doi/abs/10.5555/3495724.3496517

Ma, L., Zhang, Z., Wang, X., Li, J., & Li, G. (2023). A comprehensive survey on vector database: Storage and retrieval techniques, challenges. *arXiv preprint*. https://arxiv.org/pdf/2310.11703

Malkov, Y. A., & Yashunin, D. A. (2018). 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. https://doi.org/10.1109/TPAMI.2018.2889473

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge: Cambridge University Press. https://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf

Oliveira, L. L., Romeu, R. K., & Moreira, V. P. (2021). REGIS: A test collection for geoscientific documents in Portuguese. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 2363–2368). New York, NY: ACM. https://doi.org/10.1145/3404835.3463256

Pan, J. J., Wang, J., & Li, G. (2024). Survey of vector database management systems. *The VLDB Journal*. Berlin: Springer. https://doi.org/10.1007/s00778-024-00752-9

Radlinski, F., & Craswell, N. (2017). A theoretical framework for conversational search. In *Proceedings of the 2017 Conference on Human Information Interaction and Retrieval (CHIIR'17)* (pp. 117–126). New York, NY: ACM. https://doi.org/10.1145/3020165.3020183

Silva, J. R., & Caseli, H. M. (2021). Sense representations for Portuguese: Experiments with sense embeddings and deep neural language models. *arXiv preprint*. https://arxiv.org/abs/2109.00025

Souza, F. D., & Santos Filho, J. B. O. (2022). Embedding generation for text classification of Brazilian Portuguese user reviews: From bag-of-words to transformers. *arXiv preprint*. https://arxiv.org/abs/2212.00587

Srivastava, A. (2023). Choosing a vector database for your Gen AI stack. *SingleStoreDB Blog*. https://www.singlestore.com/blog/choosing-a-vector-database-for-your-gen-ai-stack/

Zhang, Y., Liu, S., & Wang, J. (2024). Are there fundamental limitations in supporting vector data management in relational databases? A case study of PostgreSQL. In *IEEE 40th International Conference on Data Engineering (ICDE)* (pp. 3640–3653). Utrecht: IEEE. https://doi.org/10.1109/ICDE60146