Minerva: potencializando a consciência situacional com linguagem natural usando LLMs no SisGEODEF

Carlos Magno O. Abreu[1], Madalena Lopes e Silva[1] e Ian José Agra Gomes[1,2]
[1]Centro de Análises de Sistemas Navais, Rio de Janeiro/RJ - Brasil
[2]Universidade Federal do Rio de Janeiro, Rio de Janeiro/RJ - Brasil

***Resumo*— A consolidação de dados geoespaciais de múltiplas fontes impõe desafios a sistemas relacionais tradicionais devido a alterações inopinadas na estrutura dos dados. No Sistema de Geoinformação de Defesa (SisGEODEF), adotou-se uma arquitetura híbrida, em que os dados são armazenados em colunas JSON de tabelas relacionais simplificadas. Essa solução proporciona flexibilidade e possibilita o armazenamento de dados heterogêneos sem a preocupação com a rigidez física estrutural, porém inviabiliza consultas SQL convencionais. Este trabalho propõe uma abordagem baseada em Modelos de Linguagem de Grande Porte (LLMs) que possibilita consultas aos dados em linguagem natural. Utiliza-se um processo de Geração Aumentada por Recuperação (RAG), em que um modelo interpreta e sumariza todos os dados e, posteriormente, recupera os mais relevantes a partir da consulta textual do usuário em linguagem natural. Os resultados indicam que o agente, integrado ao *chatbot* LLM do SisGEODEF (Minerva), permite consultas eficientes, intuitivas e escaláveis, mesmo em ambientes com alta heterogeneidade e baixa previsibilidade estrutural.**

***Palavras-Chave*— LLM,RAG,PostgreSQL.**

I. Introdução

A consolidação de dados geoespaciais de múltiplas fontes institucionais, como órgãos civis e militares, é uma necessidade crescente em ambientes e planejamento de Defesa. No contexto do Sistema de Geoinformação de Defesa (SisGEODEF), sistema institucional de geoinformação da área de Defesa do Brasil, a consolidação desses dados ocorre por meio de um mecanismo de interoperabilidade que agrega dados de diversas origens, cada uma com sua estrutura própria, terminologia e dinâmica de atualização.

Para minimizar os impactos no sistema causados pelas mudanças estruturais dos dados consumidos, adotou-se um modelo híbrido: embora baseado em um sistema relacional, as tabelas armazenam os dados reais em colunas do tipo JSON (*JavaScript Object Notation*) [1], de forma documental. Cada fonte possui suas próprias tabelas, mas todas compartilham uma estrutura mínima comum, com campos de controle e uma coluna JSON. Isso permite absorver alterações inopinadas na estrutura dos dados de origem sem necessidade de migração de esquema ou reestruturação de tabelas. Esse modelo oferece robustez operacional, mas impõe limitações significativas no acesso e análise dos dados: não é possível realizar *JOIN*, filtros ou agregações SQL tradicionais sem conhecer previamente a estrutura dos documentos JSON. Além disso, o número de tabelas e a heterogeneidade das fontes conferem um elevado grau de dificuldade à navegação manual e a relacionamentos convencionais.

Carlos Magno O. Abreu, magno.femar@marinha.mil.br; Madalena Lopes e Silva, madalena@marinha.mil.br; Ian José Agra Gomes, agra@marinha.mil.br.

Este artigo apresenta uma solução baseada em modelos de linguagem (LLMs) e *embeddings* semânticos [2], [3], que possibilita consultas em linguagem natural sobre esse ambiente documental heterogêneo utilizando o *chatbot* Minerva, do SisGEODEF. A proposta é dividida em duas fases: (1) modelagem semântica das tabelas com *embeddings* vetoriais a partir dos metadados de controle e (2) interpretação das consultas por um LLM para determinar as tabelas relevantes e extrair os dados diretamente dos arquivos JSON. Essa abordagem viabiliza a exploração dos dados do SisGEODEF por analistas, operadores e gestores, sem a necessidade de conhecimento técnico avançado em bancos de dados ou sobre a estrutura existente nas tabelas.

Apresentamos uma prova de conceito que avalia o método em dados institucionais reais, porém com amostra restrita, que serve de base para a extensão a múltiplas tabelas. O objetivo é demonstrar a viabilidade da metodologia e levantar requisitos para a validação ampliada no acervo heterogêneo do SisGEODEF.

II. Trabalhos Relacionados

Existem diversos estudos que utilizam Geração Aumentada por Recuperação (*Retrieval-Augmented Generation* - RAG) para auxiliar consultas em bancos de dados. O trabalho de Wu [4] propõe um *pipeline* RAG que utiliza metadados extraídos de esquemas para recuperação e geração de consultas SQL *context-aware*. Poliakov & Shvai [5] propuseram o Multi-Meta-RAG, que aprimora a filtragem semântica ao usar *embeddings* construídos a partir de metadados, reforçando a escolha de documentos relevantes em lógicas multi-hop.

III. Fundamentação Teórica

A arquitetura proposta neste trabalho combina diferentes tecnologias recentes nas áreas de ciência de dados, bancos documentais, geoinformação e inteligência artificial. Esta seção apresenta os fundamentos teóricos necessários para a compreensão da solução.

A. Sistema de Geoinformação de Defesa - SisGEODEF

O SisGEODEF foi criado por meio da Portaria GM-MD nº 2.445, de 1º de junho de 2021 [6], e institucionalizado pela Portaria Normativa nº 49/GM-MD, de 10 de julho de 2019 [7]. Seu propósito é:

- Unificar e padronizar a produção e o compartilhamento de dados geoespaciais pelas três Forças Armadas, por órgãos civis federais e pela estrutura do Ministério da Defesa;
- Criar uma Infraestrutura de Dados Espaciais de Defesa (IDE-Defesa) para sustentar operações e decisões estratégicas;
- Estabelecer o ConGEODEF (Conselho de Geoinformação de Defesa), um órgão técnico-militar interinstitucional com foco em definir normas, padrões e diretrizes relacionadas à geoinformação de defesa; e
- Apoiar o planejamento de operações militares e a logística conjunta por meio de uma base geográfica robusta, interoperável e atualizada.

O SisGEODEF oferece um conjunto de APIs (*Application Programming Interfaces*) para que outros sistemas possam consumir seus dados de forma simples, oferecendo informações cartográficas e meteorológicas de mais de 40 instituições brasileiras, incluindo o Instituto de Cartografia Aeronáutica (ICA), Banco de Dados Geográficos do Exército (BDGEx), Instituto Nacional de Meteorologia (INMET), Centro de Hidrografia da Marinha (CHM), Infraestrutura Nacional de Dados Espaciais (INDE) e o banco de dados vetoriais do OpenStreetMap (OSM). Atualmente, oferta dados geoespaciais para sistemas do Ministério da Defesa e da Marinha do Brasil. É administrado pela Coordenação-Geral de Geoinformação, Meteorologia e Aerolevantamento (CGGMA) do Ministério da Defesa.

Em relação à INDE, o diferencial do SisGEODEF é a efetiva coleta e armazenamento dos dados em seu banco de dados local, e não o apontamento para os dados originais nas fontes. Isso possibilita seu funcionamento em situações em que não há acesso às redes externas, como operações militares ou casos extremos de comprometimento da infraestrutura de comunicação.

O SisGEODEF oferece ao usuário, além de suas demais funcionalidades, um *chatbot* denominado Minerva, ilustrado na Fig. 1, que é uma interface conversacional baseada em um LLM executado localmente no servidor do sistema.

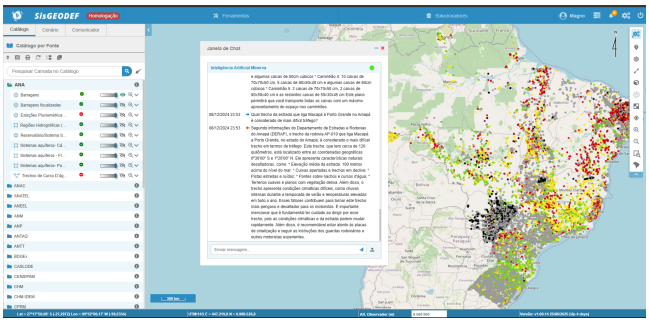


Fig. 1. O SisGEODEF com o *chatbot* LLM Minerva

B. Modelos de Linguagem de Grande Porte - LLMs

LLMs são redes neurais profundas treinadas em grandes volumes de texto para aprender padrões linguísticos, sintaxe e conhecimento factual [8]. Quando suficientemente grandes, esses modelos são capazes de realizar tarefas como resumo automático, tradução, geração de código e interpretação de consultas em linguagem natural.

Neste trabalho, utilizamos o LLM *Llama-3.2-3B-Instruct* [9], um modelo de uso geral, ajustado para seguir instruções e com grande capacidade de contexto (32k tokens), para interpretação, sumarização e geração de descrições. Utilizamos também o modelo de *embedding* *nomic-embed-text*, desenvolvido pela Nomic AI, especializado em transformar textos curtos ou médios em vetores semânticos de alta qualidade [10]. Esses vetores são utilizados para calcular similaridade e recuperar documentos relevantes por meio de buscas vetoriais.

C. Estrutura JSON e bancos documentais

Em bancos de dados relacionais modernos, como o PostgreSQL, há suporte nativo para colunas dos tipos JSON e JSONB, que permitem armazenar documentos com estrutura livre diretamente dentro de tabelas relacionais [1]. Isso oferece grande flexibilidade no armazenamento de dados heterogêneos e facilita a integração com fontes externas cuja estrutura não é rígida ou conhecida previamente.

D. Dados Geoespaciais com PostGIS

O PostGIS é uma extensão espacial do banco de dados PostgreSQL que adiciona suporte a tipos de dados geográficos e funções espaciais. Ele transforma o PostgreSQL em um Sistema Gerenciador de Banco de Dados Espacial compatível com o padrão OGC (Open Geospatial Consortium), permitindo operações como interseção, distância, buffer e consultas espaciais indexadas [11].

E. Embeddings e Vetores Semânticos

Embeddings são representações numéricas de informações (como palavras, frases ou documentos) em espaços vetoriais contínuos. Eles permitem que informações semanticamente próximas estejam próximas no espaço vetorial, mesmo que suas formas textuais sejam diferentes.

Essa representação é usada para buscar por similaridade semântica, ao invés de igualdade literal [12]. A comparação entre *embeddings* vetoriais é uma etapa essencial para recuperar informações relevantes em sistemas baseados em semântica, como o apresentado neste trabalho. A principal métrica utilizada é a distância cosseno, que mede o ângulo entre dois vetores normalizados em um espaço multidimensional. Essa métrica é particularmente eficaz em capturar relações semânticas mesmo entre textos lexicalmente diferentes, por exemplo:

- "Aeroporto" e "Pista de pouso"
- "Terra indígena" e "Área protegida"
- "Aeroporto" e "Rodovia federal"

Para controlar o nível de sensibilidade da busca, utiliza-se um limiar de corte (*threshold*). Apenas os *embeddings* com similaridade superior ao *threshold* são considerados relevantes.

F. Retrieval-Augmented Generation

A arquitetura conhecida como RAG (*Retrieval-Augmented Generation*) combina a recuperação semântica de documentos

relevantes com a geração de respostas contextualizadas por meio de modelos de linguagem. Essa abordagem se mostra eficaz em tarefas que requerem acesso a informações externas ao conhecimento estático do modelo, conforme demonstrado por Lewis et al. [2].

O processo ocorre em duas fases principais:

1) Recuperação (Retrieval): A entrada do usuário (uma pergunta ou frase) é convertida em um vetor de *embedding* e comparada a um índice vetorial que armazena representações semânticas de documentos ou objetos previamente processados. Essa etapa recupera os documentos mais similares segundo uma métrica como a distância cosseno.

2) Geração (Generation): Os documentos recuperados são passados como contexto adicional para um modelo de linguagem (LLM), que então gera uma resposta, um resumo ou uma consulta, levando em consideração tanto a entrada original quanto os documentos recuperados.

IV. Estrutura dos Dados no SisGEODEF

O SisGEODEF possui um módulo coordenador de interoperabilidade, encarregado de conectar com os servidores das fontes oficiais de geoinformação, coletar e armazenar os dados em seu banco de dados interno. Devido à natureza extremamente heterogênea destes dados, à quantidade de fontes diferentes e às constantes mudanças estruturais, não é possível criar tabelas com estruturas rígidas relacionais que os representem. Desta forma, foi adotado um modelo híbrido, onde todas as tabelas que armazenam os dados das fontes (atualmente são 660 tabelas) possuem exatamente a mesma estrutura física no banco de dados, com campos de controle como: atributos, idfonte e geom, conforme ilustrado na Fig. 2. No exemplo apresentado, os atributos do dado, como número da pista, número de faixas ou população, estão encapsulados no campo JSON denominado “atributos”, cujo conteúdo e estrutura podem ser modificados pelo órgão fornecedor sem aviso prévio e sem prejudicar o armazenamento.

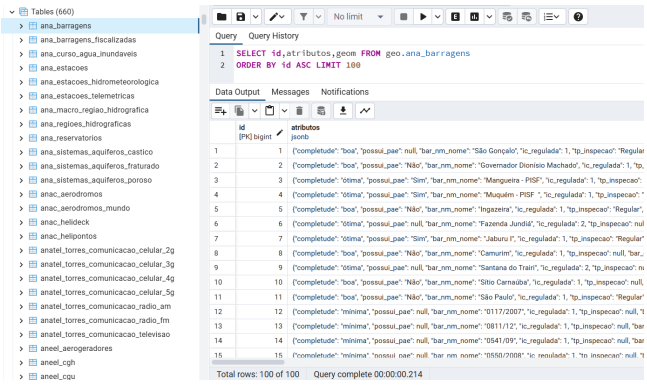


Fig. 2. Tabelas de dados do SisGEODEF

Uma tabela é utilizada adicionalmente para armazenar o cadastro de fontes de dados. Sendo assim, todos os registros em cada tabela de dados possuem um campo “idfonte” que faz a relação com o registro correspondente na tabela de fontes, conforme ilustrado na Fig. 3.

Essa abordagem, embora viabilize o armazenamento e atualização dos dados no sistema sem impacto na infraestrutura, impede a consulta aos dados de forma eficiente, levando aos seguintes desafios:

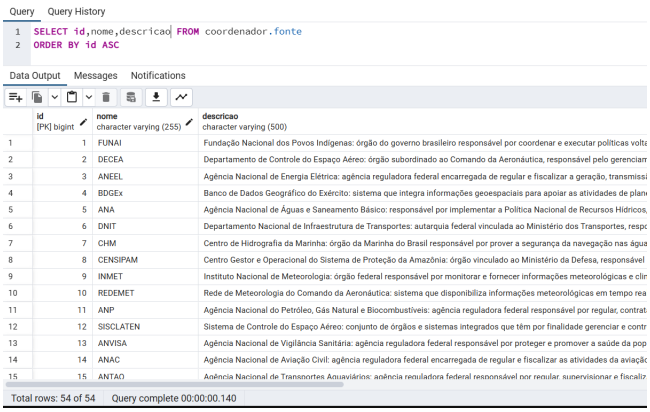


Fig. 3. Tabelas de fontes do SisGEODEF

- Impossibilidade de normalização relacional: inviável criar tabelas específicas por fonte, pois as fontes produtoras não fornecem documentação ou não possuem compromisso com a estrutura dos dados, apenas com o conteúdo.
- Estrutura não previsível: os órgãos produtores alteram as estruturas dos dados fornecidos sem qualquer aviso.
- Número elevado de tabelas: a quantidade de tabelas de armazenamento dos dados geoespaciais está diretamente ligada às camadas geoespaciais que cada fonte oferece e pode aumentar ou diminuir de acordo com as necessidades do Ministério da Defesa.
- Conjunto de dados sujeito à alterações frequentes: o Agente Coordenador do SisGEODEF atualiza semanalmente dezenas de tabelas e centenas de registros, tornando inviável o treinamento de um modelo específico que contenha o conhecimento armazenado no banco de dados.

V. Arquitetura Proposta

A arquitetura proposta, ilustrada na Fig. 4, foi desenvolvida para permitir consultas semânticas em linguagem natural sobre um banco de dados relacional documental heterogêneo, com dados armazenados no formato JSON. A implementação ocorre majoritariamente na linguagem Java, que orquestra as interações entre o banco de dados, o servidor local de modelos (Ollama) e a interface de consulta do usuário via chatbot Minerva.

Enquanto muitos trabalhos sobre RAG que utilizam modelos LLM operam em datasets sintéticos, exemplos simples ou protótipos isolados, a solução apresentada neste trabalho foi utilizada em um contexto institucional real e crítico, o SisGEODEF, com dados volumosos e não normalizados, o qual consome dados fornecidos por instituições tais como Departamento Nacional de Infraestrutura de Transportes (DNIT), Fundação Nacional dos Povos Indígenas (FUNAI), Banco de Dados Geográficos do Exército (BDGEX), Departamento de Controle do Espaço Aéreo (DECEA), dentre outros.

No contexto deste trabalho, a técnica RAG é aplicada para recuperar descrições de objetos JSON oriundos das tabelas do SisGEODEF. A geração subsequente produz resumos explicativos sobre os registros encontrados. Essa abordagem permite contornar a limitação dos LLMs de não possuírem acesso ao banco de dados em tempo real, ao mesmo tempo

em que mantém a flexibilidade de uma consulta semântica robusta.

Ao usar o servidor Ollama local, a presente proposta é completamente desacoplada de serviços externos (como OpenAI, Anthropic ou Google Cloud), garantindo segurança, confidencialidade e controle institucional, requisitos extremamente relevantes em ambientes militares, onde o controle de acesso e soberania da informação são essenciais.

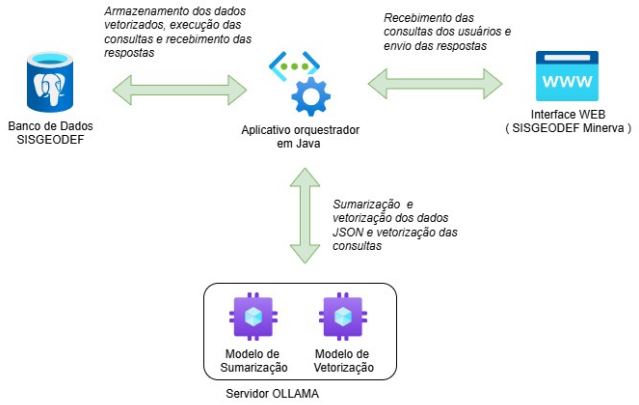


Fig. 4. Arquitetura proposta

Todos os modelos utilizados são executados localmente por meio do servidor Ollama, que atua como um orquestrador de modelos LLM em ambiente local. O Ollama permite carregar, isolar e executar diferentes tipos de modelos (de código, texto, *embeddings*, etc.) com alta performance e sem necessidade de conexão com APIs externas, garantindo segurança e soberania da informação institucional.

O presente trabalho também usa um LLM para interpretar o JSON e gerar uma descrição textual otimizada antes de gerar *embeddings*, que não apenas vetoriza dados brutos, mas os condensa semanticamente antes da indexação. Essa etapa de pré-sumarização vetorial é raramente explorada em *pipelines* RAG tradicionais, além de contar com um mecanismo de tolerância semântica adaptativa, com controle refinado com base em um *threshold* de proximidade vetorial, facilitando ajustes em casos de uso diferentes.

No contexto do SisGEODEF, cada tabela de dados possui uma estrutura fixa mínima (campos de controle), e uma coluna atributos que armazena os dados reais em JSON. Essa abordagem favorece a flexibilidade e a importação de fontes heterogêneas sem necessidade de remodelagem do banco. Um programa em Java percorre todas as tabelas de dados, acessando o conteúdo dos campos JSON de cada registro. Esses dados são enviados a um modelo LLM de uso geral Llama 3.2, modificado para suportar 32K de memória de contexto, para geração de uma descrição textual curta e objetiva, que sintetiza semanticamente o conteúdo de cada objeto. Essa descrição é então transformada em um vetor semântico por meio do modelo *nomic-embed-text*, e armazenada em uma tabela própria no PostgreSQL, com campo do tipo *vector* utilizando-se a extensão *pgvector*. Cada vetor gerado é vinculado a um identificador único que referencia o registro original na tabela de origem.

Mesmo com os dados principais armazenados em JSON, a estrutura básica das tabelas do SisGEODEF inclui um campo geométrico padrão, possibilitando interações geoespaciais em um mapa. Como todas as informações obtidas pelo SisGEODEF possuem a geometria geoespacial em comum, este é o único método possível de relacionamento convencional entre os dados no banco.

Quando o usuário realiza uma consulta em linguagem natural através da interface Minerva, o texto da pergunta é vetorizado pelo mesmo modelo de *embeddings*. Em seguida, o vetor resultante é comparado com os *embeddings* previamente armazenados no banco, utilizando busca por similaridade calculada pela distância cosseno com limiar configurável. Os dez resultados mais próximos são apresentados na interface para seleção manual. Uma vez que o usuário identifica o item desejado, o sistema utiliza o identificador associado para recuperar o registro original e exibir sua representação geoespacial no mapa do SisGEODEF.

Devido a limitações computacionais no uso de modelos LLM, uma única tabela com 21.891 registros foi utilizada. O objetivo foi avaliar a recuperação, pelo modelo LLM, de informações previamente conhecidas pelo operador e avaliar a aplicação correta do contexto no qual os dados foram solicitados.

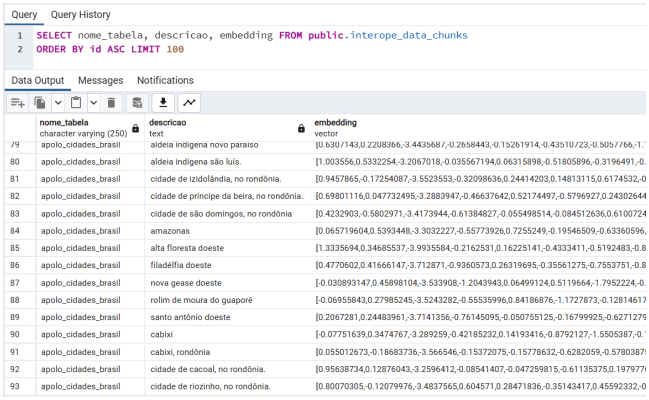
A tabela onde são armazenados os vetores, conforme ilustrado na Fig. 5, mantém um registro que aponta para o dado original. O objetivo é manter uma estrutura separada da estrutura do próprio SisGEODEF. É nesta tabela que o modelo irá realizar a busca por similaridade semântica com o texto da consulta do usuário.

No sistema implementado, foi utilizado inicialmente um *threshold* de 0.6005, valor empiricamente ajustado para equilibrar *recall* (abrangência) e precisão (relevância). Valores altos (acima de 0.8) garantem maior precisão, mas podem deixar de recuperar dados vagamente relacionados, enquanto valores baixos (menores que 0.5) podem recuperar mais resultados, mas aumentam a chance de ruído ou irrelevância.

A escolha do valor do limiar de corte depende do contexto: consultas muito genéricas exigem maior tolerância, enquanto buscas técnicas demandam precisão. No caso do SisGEODEF, onde termos similares podem ter origens variadas (ex: “aldeia”, “reserva”, “terra tradicional”), o *threshold* permite controlar o quanto de proximidade semântica será aceito antes de apresentar resultados ao usuário. Além disso, a cartografia básica possui algumas variações nas denominações de alguns elementos. Por exemplo, uma rocha submersa pode ser classificada como “obstáculos submersos”, “formações rochosas subaquáticas”, “rocha submersa”, “pedra submersa” ou “rochas encobertas”, dependendo se o dado foi gerado pela Diretoria do Serviço Geográfico do Exército (DSG) ou a Diretoria de Hidrografia e Navegação da Marinha (DHN). Porém, para o usuário final, tudo o que ele pode estar procurando pode se resumir a uma pedra embaixo da água. A qualidade dos dados também afeta diretamente a escolha do valor apropriado para o limiar de corte. Dados muito pobres tendem a gerar sumarizações mais genéricas, necessitando de mais tolerância.

Foram utilizados os seguintes modelos LLM:

- *embeddings* semânticos: *nomic-embed-text*, responsável por transformar descrições de tabelas e consultas em vetores para busca vetorial.
- Modelos de uso geral e sumarização: *llama3.2-ctx-32768*, usado para tarefas genéricas como gerar descrições de tabelas e interpretar intenções do usuário



Query Query History

```
SELECT nome_tabela, descricao, embedding FROM public.interope_data_chunks
ORDER BY id ASC LIMIT 100
```

Data Output Messages Notifications

	nome_tabela character varying (250)	descricao text	embedding vector
79	apolo_cidades_brasil	aldeia indígena novo paraíso	[0.630743,0.2208366,-3.4435687,-0.2668443,0.15261914,-0.43510723,-0.5057766,-1.
80	apolo_cidades_brasil	aldeia indígena são luís.	[1.003556,0.5332254,-3.2067018,-0.035567194,0.06315898,-0.51805896,-0.3196491,-0.
81	apolo_cidades_brasil	cidade de izidolândia, no rondônia.	[0.9457865,-0.17254087,-3.5523553,-0.32098636,0.24414203,0.14813115,0.6174532,-0
82	apolo_cidades_brasil	cidade de príncipe da beira, no rondônia.	[0.69801116,0.047732495,-3.2883947,-0.46637642,0.52174497,-0.5796927,0.24302644
83	apolo_cidades_brasil	cidade de são domingos, no rondônia	[0.4232903,-0.5802971,-3.4173944,-0.61384827,-0.055498514,-0.084512636,0.6100724
84	apolo_cidades_brasil	amazonas	[0.065719604,0.5393448,-3.3032227,-0.55773926,0.7255249,-0.19546509,-0.63360596,
85	apolo_cidades_brasil	alta floresta doeste	[1.3335694,0.34685537,-3.9935584,-0.2162531,0.16225141,-0.4333411,-0.5192483,-0.8
86	apolo_cidades_brasil	filadélfia doeste	[0.4770602,0.41666147,-3.712871,-0.9360573,0.26319695,-0.35561275,-0.7553751,-0.8
87	apolo_cidades_brasil	nova gease doeste	[-0.030893147,0.45898104,-3.533908,-1.2043943,0.06499124,0.5119664,-1.7952224,-0.
88	apolo_cidades_brasil	rolim de moura do guaporé	[-0.06955843,0.27985245,-3.5243282,-0.55535996,0.84186876,-1.1727873,-0.12814617
89	apolo_cidades_brasil	santo antônio doeste	[0.2067281,0.24483961,-3.7141356,-0.76145095,-0.050755125,-0.16799925,-0.6271279
90	apolo_cidades_brasil	cabixi	[-0.07751639,0.3474767,-3.289259,-0.42185232,0.14193416,-0.8792127,-1.5505387,-0.
91	apolo_cidades_brasil	cabixi, rondônia	[0.055012673,-0.18683736,-3.566546,-0.15372075,-0.15778632,-0.6282059,-0.5780387
92	apolo_cidades_brasil	cidade de cacoal, no rondônia.	[0.95638734,0.12876043,-3.2596412,-0.08541407,-0.04725981,-0.61135375,0.197977
93	apolo_cidades_brasil	cidade de riozinho, no rondônia.	[0.80070305,-0.12079976,-3.4837565,0.604571,0.28471836,-0.35143417,0.45592332,-0

Fig. 5. Tabela de *embeddings*

de forma contextualizada.

Foi necessário utilizar modelos de baixa capacidade devido às limitações computacionais do ambiente.

O ambiente de execução foi composto de:

- Banco de dados: PostgreSQL utilizando PostGIS com tabelas contendo dados geoespaciais em colunas do tipo JSON. O conjunto de dados utilizado foi o do próprio servidor de testes do SisGEODEF.
- Servidor local de modelos: Ollama configurado com múltiplos modelos especializados.
- Cliente Java: responsável por orquestrar as chamadas aos modelos, gerenciar *embeddings* e executar consultas.

VI. Resultados e Discussão

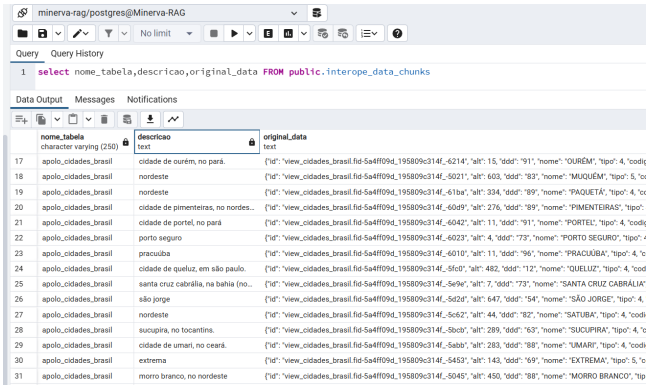
A. Criação dos dados de teste

Para os dados de teste, foi selecionada uma tabela de dados do SisGEODEF contendo cidades do Brasil. Os dados JSON desta tabela foram submetidos ao modelo de sumarização para interpretação e depois foram gerados os *embeddings*.

O modelo sumarizou os dados utilizando o seguinte prompt:

```
Descreva de forma objetiva o objeto JSON a
seguir. Seja breve. Sem explicações.
Não considere atributos nulos.
```

O resultado da sumarização, ilustrado na Fig. 6, foi, na grande maioria dos casos, no formato "Cidade de Niterói, no Rio de Janeiro". Alguns resultados variaram de acordo com o conteúdo do objeto JSON, o que eventualmente incluía o número de habitantes ou área total urbana.



minerva-rag/postgres@Minerva-RAG

Query Query History

```
select nome_tabela,descricao,original_data FROM public.interope_data_chunks
```

Data Output Messages Notifications

	nome_tabela character varying (250)	descricao text	original_data text
17	apolo_cidades_brasil	cidade de ourém, no pará.	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-6214", "alt": 15, "ddd": "91", "nome": "OURÉM", "tipo": 4, "codi
18	apolo_cidades_brasil	nordeste	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-5021", "alt": 603, "ddd": "83", "nome": "MUQUÉM", "tipo": 5, "co
19	apolo_cidades_brasil	nordeste	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-61ba", "alt": 334, "ddd": "89", "nome": "PAQUETÁ", "tipo": 4, "co
20	apolo_cidades_brasil	cidade de pimenteiras, no nordes...	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-60d9", "alt": 276, "ddd": "89", "nome": "PIMENTEIRAS", "tipo":
21	apolo_cidades_brasil	cidade de portel, no pará	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-6042", "alt": 11, "ddd": "91", "nome": "PORTEL", "tipo": 4, "codi
22	apolo_cidades_brasil	porto seguro	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-6023", "alt": 4, "ddd": "73", "nome": "PORTO SEGURO", "tipo": 4
23	apolo_cidades_brasil	pracuúba	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-6010", "alt": 11, "ddd": "96", "nome": "PRACUÚBA", "tipo": 4, "c
24	apolo_cidades_brasil	cidade de queluz, em são paulo.	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-5fc0", "alt": 482, "ddd": "12", "nome": "QUELUZ", "tipo": 4, "cod
25	apolo_cidades_brasil	santa cruz cabrália, na bahia (no...	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-5e9e", "alt": 7, "ddd": "73", "nome": "SANTA CRUZ CABRÁLIA"
26	apolo_cidades_brasil	são jorge	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-5d2d", "alt": 647, "ddd": "54", "nome": "SÃO JORGE", "tipo": 4,
27	apolo_cidades_brasil	nordeste	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-5c62", "alt": 44, "ddd": "82", "nome": "SATUBA", "tipo": 4, "codi
28	apolo_cidades_brasil	sucupira, no tocantins.	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-5bcb", "alt": 289, "ddd": "63", "nome": "SUCUPIRA", "tipo": 4, "c
29	apolo_cidades_brasil	cidade de umari, no ceará.	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-5abb", "alt": 283, "ddd": "88", "nome": "UMARI", "tipo": 4, "codi
30	apolo_cidades_brasil	extrema	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-5453", "alt": 143, "ddd": "69", "nome": "EXTREMA", "tipo": 5, "c
31	apolo_cidades_brasil	morro branco, no nordeste	{"id": "view_cidades_brasil.fid-5a4ff09d_195809c314f_-5045", "alt": 450, "ddd": "88", "nome": "MORRO BRANCO", "tip

Fig. 6. Tabela de sumarização (*embeddings*) (parcial)

B. Avaliação

As solicitações de consulta e a obtenção das respostas foram injetadas e recebidas, respectivamente, através da linha de comando utilizando o software "curl", que fez as funções de interface com o usuário ao se comunicar com o Minerva.

Os testes foram conduzidos manualmente, onde alguns usuários do SisGEODEF com conhecimentos em geoinformação e na natureza dos dados armazenados no banco criaram algumas consultas relacionadas aos dados da tabela selecionada para o teste.

Durante o processo, foi necessário realizar um ajuste fino do *threshold*, evitando os extremos onde não retornava resultados mesmo quando a consulta incluía expressões exatamente iguais à sumarização ou quando retornava resultados mesmo quando a consulta era totalmente divergente dos dados armazenados. Um valor de cerca de 60% de similaridade (0,6005) foi considerado satisfatório, mas percebeu-se que este limiar deve ser ajustado adequadamente de acordo com a qualidade dos dados sumarizados. Assim, quanto melhor a qualidade do dado, menos permissivo o valor poderia ser.

Para testar a confiabilidade das respostas, decidiu-se pesquisar primeiro por textos exatamente iguais aos que já se sabia existir no banco. Sendo assim, foram selecionados 10 registros diferentes na tabela de *embeddings* e utilizados seus textos sumarizados como consultas. O texto sumarizado corresponde ao dado vetorizado do mesmo registro. Desta forma, ao se gerar uma consulta contendo exatamente o texto sumarizado, o registro retornado deverá ser invariavelmente aquele que possui o vetor de *embeddings* com máxima similaridade semântica, quando os vetores da consulta forem extremamente próximos aos vetores do resultado. Todos os resultados das consultas retornaram os registros esperados como a primeira opção, demonstrando que o sistema estava interpretando corretamente as consultas e correlacionando-as com os vetores existentes no banco de dados.

O experimento seguinte consistiu em variar a semântica da consulta, simulando usuários que não sabem exatamente o que procurar ou aqueles que lembram parcialmente da informação desejada. O experimento também simulou situações onde os usuários forneciam consultas truncadas ou com erros de ortografia e/ou acentuação.

Com a finalidade de simplificação, como a metodologia e os resultados obtidos foram os mesmos para todos os 10 registros selecionados, este trabalho irá considerar apenas um deles: o registro da cidade de Príncipe da Beira, em Rondônia. Foram criadas variações desta expressão, que iam desde sua representação exata até expressões que se aproximavam vagamente da original. As consultas realizadas estão elencadas na Tabela I.

TABELA I

Consultas realizadas na tentativa de encontrar a cidade de Príncipe da Beira - RO

Texto
"cidade de príncipe da beira, rondônia"
"Procure pela cidade príncipe, não lembro o resto do nome"
"principe da beira"
"cidade da beira em rondônia"
"principe de Rondônia"
"principi de birra"
"Privepe da beirra de rondinia"

C. Discussão

Nas consultas de correspondência literal e em variações ortográficas simples sobre a tabela avaliada, o item-alvo apareceu de forma consistente nas primeiras posições (top-1) dos resultados. Essa evidência é restrita ao cenário de uma única tabela e não permite inferir desempenho global em todo o acervo heterogêneo do SisGEODEF; portanto, evitamos generalizações além do escopo deste protótipo.

Entretanto, ao utilizar outros parâmetros de busca, como por exemplo, número de habitantes ou área, o modelo ficou dependente da qualidade dos dados fornecidos e de como ele conseguiu sumarizar os dados JSON na fase inicial ao interpretar os objetos JSON. Esta capacidade está ligada diretamente à assertividade do prompt utilizado.

Uma vez que se possa definir um valor de *threshold* adequado para um bom grau de proximidade entre o dado encontrado e o esperado, é possível rastrear o dado original e obter o objeto JSON que possibilitou a consulta, entregando ao usuário as respostas 100% assertivas. A Tabela II mostra a lista de resultados para uma consulta, ordenada por similaridade semântica.

TABELA II

RESULTADOS POR ORDEM DE SIMILARIDADE

Similaridade	Resultado
0.66906282	cidade de príncipe da beira, rondônia
0.65369258	cidade de capiá da igrejinha, nordeste
0.64500913	cidade de são paulinho, no ceará
0.63567581	cidade de lagoa grande, maranhão
0.63427299	cidade de pimenteiras, no nordeste
0.63296204	cidade de inhamuns, no ceará
0.63157626	cidade de umari, no ceará
0.63049240	cidade de imperatriz, no maranhão
0.62918519	cidade de praia do apeu, no pará
0.62044930	cidade de espigão doeste, rondônia

D. Limitações

(i) **Escopo da validação**: a avaliação concentrou-se em uma única tabela por restrições computacionais, não cobrindo integralmente o cenário de centenas de tabelas heterogêneas; (ii) **Qualidade dos dados**: descrições geradas a partir de objetos JSON pobres podem induzir sumarizações genéricas, afetando o *threshold* ótimo; (iii) **Sensibilidade ao *threshold***: valores distintos favorecem precisão ou abrangência; (iv) **Tipos de consulta**: o protótipo foca recuperação semântica de registros e ainda não implementa agregações/joins complexos, que serão tratadas em trabalho futuro.

VII. CONCLUSÕES E PERSPECTIVAS FUTURAS

O uso de modelos distintos e especializados em cada parte do processo permitiu um bom grau de precisão, mesmo diante da alta variabilidade estrutural dos dados em JSON. A execução local dos modelos com o servidor Ollama viabilizou um ambiente seguro e reprodutível para uso institucional, sem a dependência de serviços online de terceiros. Entretanto, a falta de um ambiente computacional adequado comprometeu significativamente a condução do experimento ao forçar o uso de modelos de baixa capacidade. A definição do limiar de corte foi relacionada com a qualidade dos dados sumarizados, que por sua vez estava dependente diretamente da qualidade dos dados fornecidos, como era esperado. Os dados originais na tabela escolhida (Cidades do Brasil) do SisGEODEF foram considerados muito pobres, gerando sumarizações que muitas vezes não explicavam o dado em sua plenitude, causando ambiguidades e forçando a adoção de um limiar de corte muito alto (mais restritivo). Considerando estas observações e as limitações de *hardware*, o método apresentado se mostrou satisfatório e promissor, podendo ser empregado como ferramenta poderosa quando for necessário efetuar consultas em bancos de dados com dados não normalizados, tabelas com estrutura complexa demais para criar consultas pré-definidas em código, quando existir a possibilidade de o usuário não ter pleno conhecimento do conteúdo armazenado ou quando for conveniente permitir ao usuário se comunicar com o banco de dados de forma mais fluida e natural.

Este trabalho demonstrou a viabilidade de aplicar modelos de linguagem e *embeddings* semânticos à consulta de dados documentais em ambientes operacionais como o SisGEODEF. A abordagem permitiu superar limitações impostas por esquemas não normalizados e estruturas JSON imprevisíveis.

Como trabalho futuro imediato, conduziremos uma validação com múltiplas tabelas e fontes (amostra estratificada do acervo), contemplando consultas de desambiguação, analíticas/comparativas e cenários sem resultado, com métricas de recuperação e análise de erros. Essa etapa visa verificar a generalização do método ao desafio central de heterogeneidade entre fontes e consolidar evidências para submissão a periódicos e desdobramentos acadêmicos.

REFERÊNCIAS

[1] R. Ibrahim, N. Alhadi, H. Mamat, and A. Hamdan, "A Review on JSON Data Storage and Querying," *Journal of Computer Science*, vol. 12, no. 3, pp. 121–130, 2016.

[2] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 9459–9474, 2020.

[3] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pp. 3982–3992, 2019.

[4] Z. Wu, Z. Li, J. Zhang, M. Li, Y. Zhao, R. Fang, Z. He, X. Li, Z. Li, and S. Song, "Rb-sql: A retrieval-based llm framework for text-to-sql," 2024, acesso em: 23 ago. 2025. [Online]. Available: https://arxiv.org/abs/2407.08273

[5] M. Poliakov and N. Shvai, *Multi-Meta-RAG: Improving RAG for Multi-hop Queries Using Database Filtering with LLM-Extracted Metadata*. Springer Nature Switzerland, 2025, p. 334–342. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-81372-6_25

[6] M. da Defesa (Brasil), "Portaria GM-MD nº 2.445, de 1º de junho de 2021," Diário Oficial da União, Brasília, DF, 2021, institui o Sistema de Geoinformação de Defesa (SisGEODEF) e dá outras providências.

[7] ——, "Portaria Normativa nº 49/GM-MD, de 10 de julho de 2019," Diário Oficial da União, Brasília, DF, 2019, estabelece diretrizes para a estruturação do Sistema de Geoinformação de Defesa (SisGEODEF).

[8] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan *et al.*, "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[9] Meta, "Llama 3.2 model card," Meta, Tech. Rep., 2024.

[10] N. AI, "nomic-embed-text: Open Embeddings for Text," https://docs.nomic.ai/embedding/, 2023, acessado em 17 jun. 2025.

[11] R. R. Inc., *PostGIS 3.3 Documentation*, https://postgis.net/documentation/, 2023, extensão espacial do PostgreSQL para dados geográficos.

[12] D. Dominguete, "Construindo espaços semânticos para aplicações de processamento de linguagem natural," Medium blog post on Power Through Connections, Sep. 2019. [Online]. Available: https://medium.com/power-through-connections/construindo-espaços-semânticos-para-aplicações-de-processamento-de-linguagem-natural-9dbc3dd06dd7