Laboratório de Engenharia de Sistemas e Robótica

UFPB

Modelos de Linguagem Larga (LLM)

4. RAG

Prof.: Alisson Brito (alisson@ci.ufpb.br)

Laboratório de Engenharia de Sistemas e Robótica

UFPB

Introdução

O RAG (*Retrieval-Augmented Generation*) é uma técnica que combina a **recuperação de informações** (retrieval) com a **geração de texto** (generation) para **melhorar a precisão e relevância das respostas** geradas por LLMs.

Embora os modelos possam responder algumas perguntas corretamente, eles também respondem com muita confiança a muitas perguntas de forma incorreta.

O principal método que a indústria adotou para corrigir esse comportamento é o RAG

RAG permite que o modelo recupere informações de uma base de dados ou documento relevante antes de gerar uma resposta. Isso torna as respostas mais precisas e contextuais.

Introdução

Surgiu da necessidade de superar **limitações das LLMs tradicionais**, que:

- Têm conhecimento restrito ao período de treinamento;
- Podem gerar informações incorretas ou inventadas (alucinações);
- Não conseguem acessar bases externas em tempo real.

A RAG chamou a atenção dos desenvolvedores de IA generativa pela primeira vez após a publicação de “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”. Um artigo de 2020 publicado por Patrick Lewis e uma equipe do Facebook AI Research.

Link para o artigo: https://arxiv.org/abs/2005.11401

Desde então, o conceito de RAG foi adotado por muitos pesquisadores acadêmicos e da indústria, que o veem como uma forma de **melhorar significativamente o valor dos sistemas de IA generativa.**

RAG Retrieval-Augmented Generation

O processo envolve três etapas principais:

1. **Consulta do usuário**
   O usuário faz uma pergunta ou solicita uma informação específica.

2. **Recuperação (Retrieval)**
   Um mecanismo de busca identifica documentos ou trechos relevantes em uma base de dados externa, como PDFs, artigos, FAQs, wikis ou bancos corporativos.

3. **Geração (Generation)**
   A LLM recebe o contexto recuperado e utiliza essas informações para gerar a resposta final, fundamentada e coerente.

Dense retrieval (Técnica de recuperação)

- Atua na **etapa 2 (Recuperação)**.
- Transforma a consulta e os documentos em **vetores (embeddings)** e usa busca vetorial para encontrar os documentos mais próximos semanticamente.
- Não gera respostas, apenas **recupera conteúdo relevante**.

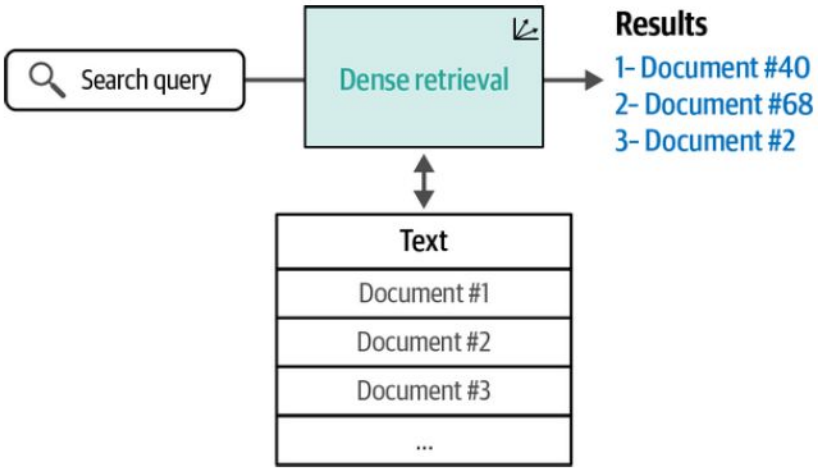


Figure 8-1. Dense retrieval is one of the key types of semantic search, relying on the similarity

Dense retrieval (Técnica de recuperação)

- Cada texto é convertido em um vetor (embedding) que representa seu significado em um espaço, onde textos próximos são semanticamente similares;
- Quando o usuário faz uma consulta, ela também vira um vetor e o sistema busca os textos mais próximos para encontrar respostas relevantes;
- Para isso, a base de conhecimento é dividida em pedaços menores, transformados em vetores e armazenados em um banco vetorial, que permite buscas rápidas e eficientes

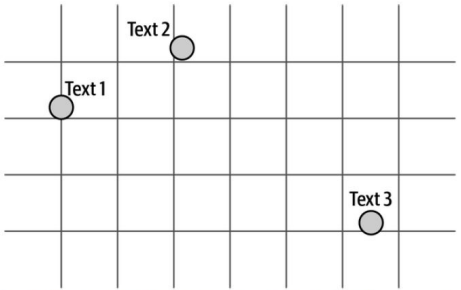


Figure 8-4. The intuition of embeddings: each text is a point and texts with similar meaning are close to each other.

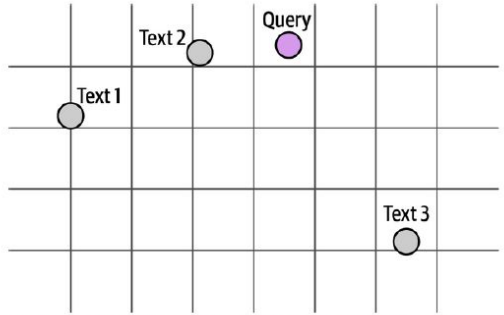


Figure 8-5. Dense retrieval relies on the property that search queries will be close to their relevant results.

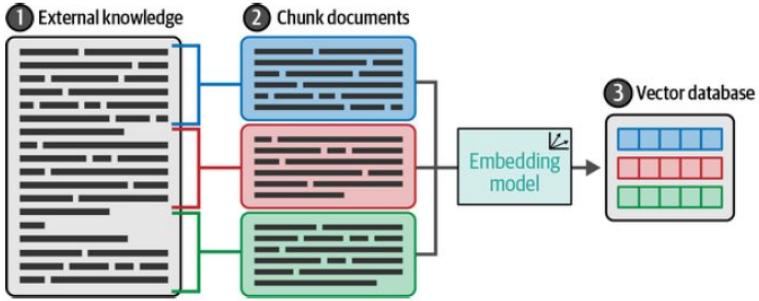


Figure 8-6. Convert some external knowledge base to a vector database. We can then query this vector database for information about the knowledge base.

Reranking (Técnica de reclassificação)

- Também atua na **etapa 2**, mas como uma **etapa refinadora** dentro da recuperação.
- Recebe uma lista de documentos pré-selecionados (ex: por dense retrieval) e os **reordena** com base na **relevância para a consulta**, usando um modelo de linguagem.
- Melhora a qualidade dos documentos que serão entregues à próxima etapa.

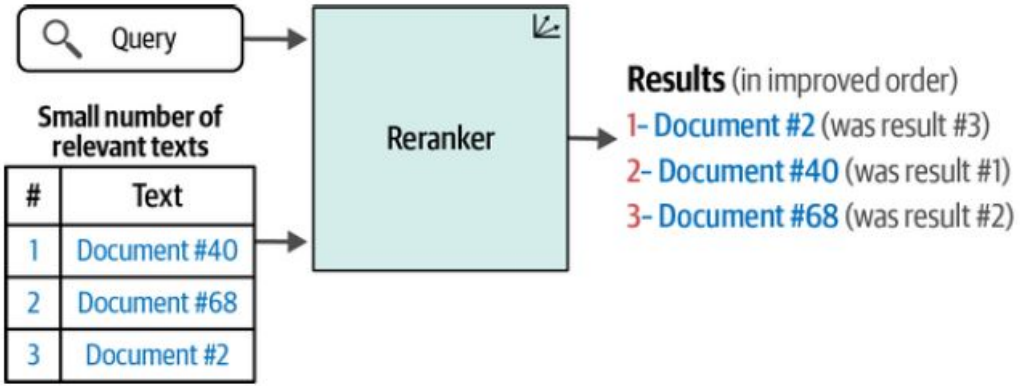


Figure 8-2. Rerankers, the second key type of semantic search, take a search query and a collection of results, and reorder them by relevance, often resulting in vastly improved results.

Reranking (Técnica de reclassificação)

- Estrutura de um sistema de busca com reclassificação, atuando como a segunda etapa em um sistema de busca em duas fases.

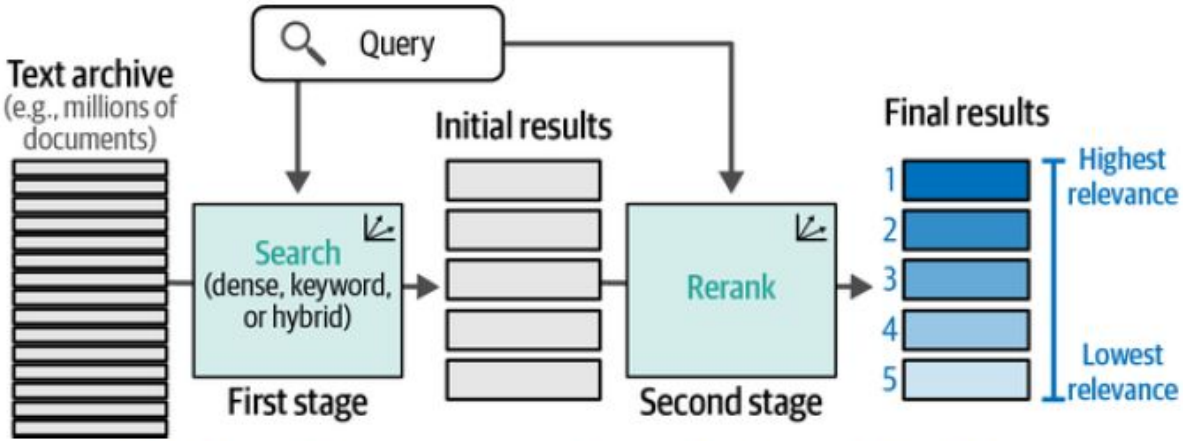


Figure 8-14. LLM rerankers operate as part of a search pipeline with the goal of reordering a number of shortlisted search results by relevance.

RAG Retrieval-Augmented Generation

Combina as etapas 2 e 3: recuperação + geração.

Primeiro, recupera-se os documentos relevantes (com dense retrieval, BM25, etc.)

Em seguida, passa esses documentos para um LLM, que gera uma resposta contextualizada e fundamentada;

A RAG existe para reduzir alucinações, melhorar factualidade, e limitar a geração a conteúdos confiáveis.

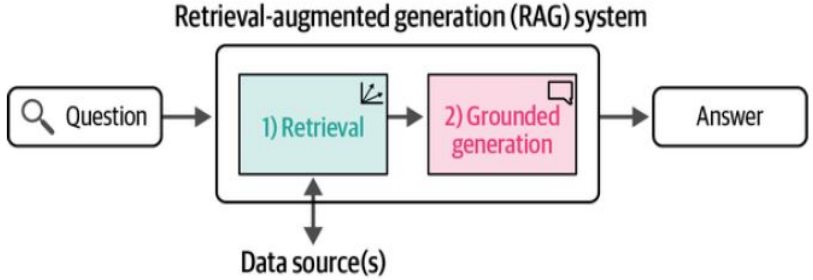


Figure 8-24. A basic RAG pipeline is made up of a search step followed by a grounded generation step where the LLM is prompted with the question and the information retrieved from the search step.

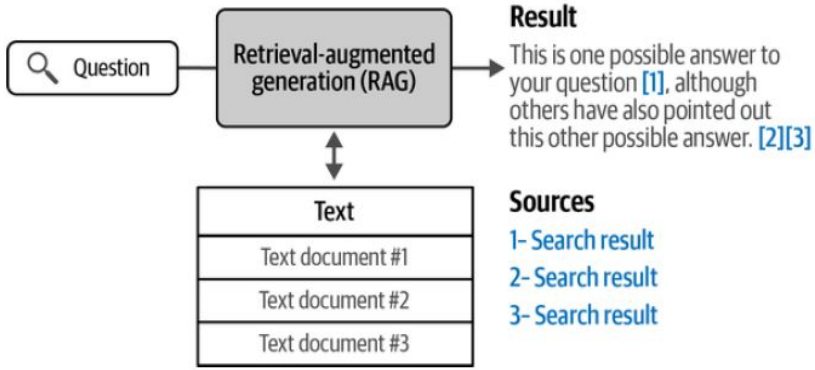
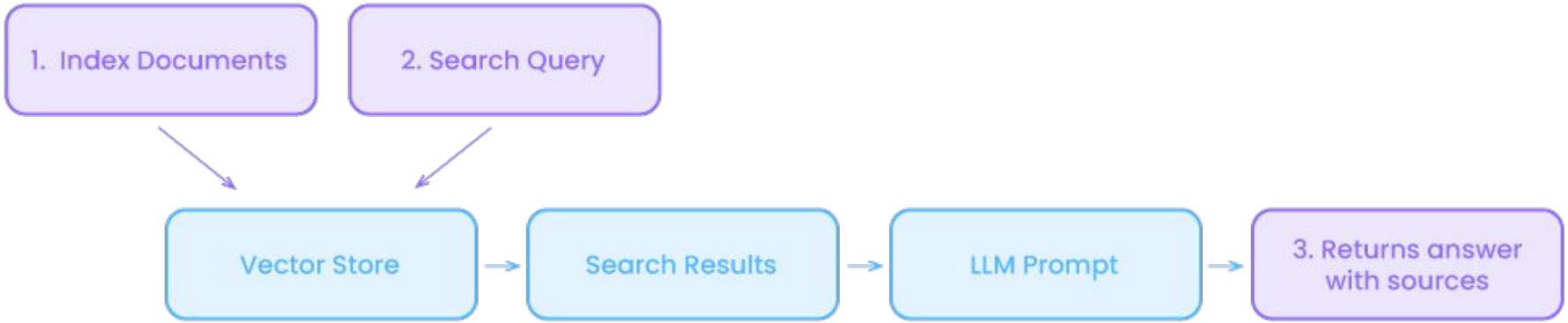


Figure 8-3. A RAG system formulates an answer to a question and (preferably) cites its information sources.

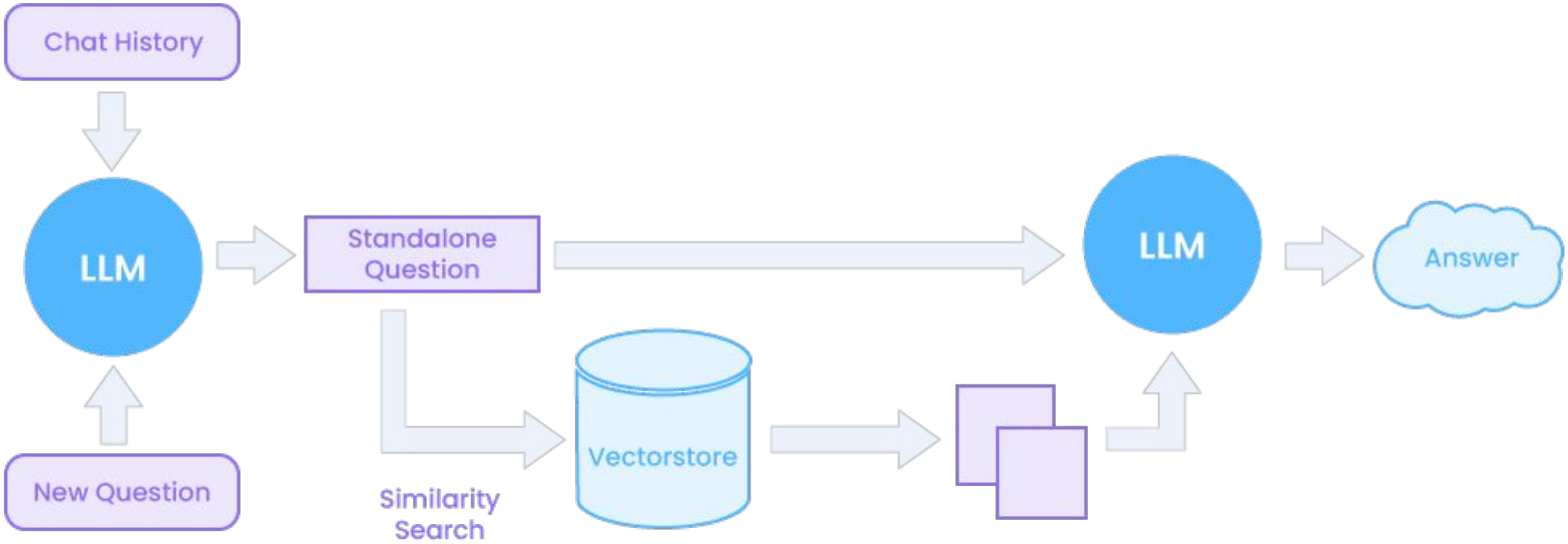
RAG Retrieval-Augmented Generation

RAG combina a capacidade de encontrar informações relevantes (recuperação), interpretá-las e complementá-las (aumento) e gerar respostas completas e bem formuladas (geração)



RAG Retrieval-Augmented Generation

Com o RAG podemos recuperar fatos de uma base de conhecimento externa para fundamentar grandes modelos de linguagem (LLMs) nas informações mais precisas e atualizadas e para fornecer aos usuários uma visão sobre o processo gerador de LLMs.



PROCESSO DE FUNCIONAMENTO

Usuário → Query → Busca Vetorial → Contexto Recuperado → LLM → Resposta Enriquecida

- **Usuário:** faz a pergunta ou solicita informação.
- **Query:** a pergunta é convertida em um vetor numérico (embedding).
- **Busca Vetorial:** encontra trechos de documentos semanticamente próximos à query.
- **Contexto Recuperado:** são os trechos relevantes usados para melhorar a resposta.
- **LLM:** gera a resposta combinando o contexto recuperado com seu conhecimento.
- **Resposta Enriquecida:** resposta final, mais precisa e fundamentada, entregue ao usuário.

PROCESSO DE FUNCIONAMENTO

Quando os usuários fazem uma pergunta a um LLM, o modelo de IA envia a consulta para outro modelo que a converte em um formato numérico para que as máquinas possam lê-la. A versão numérica da consulta às vezes é chamada de embedding ou vetor.

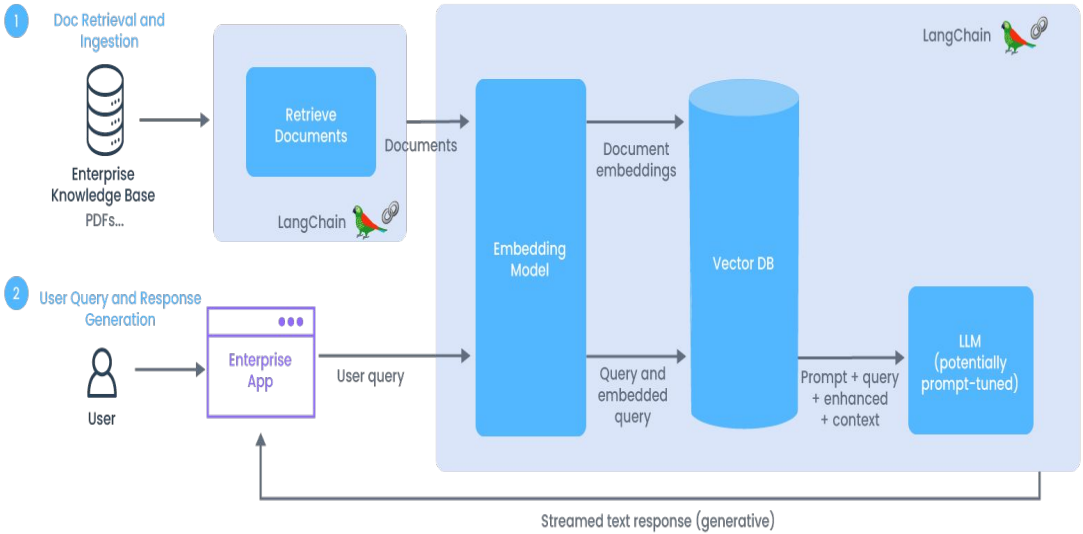
1. **Consulta**: Usuário faz a pergunta (Query)

2. **Recuperação: Dense Retrieval** encontra documentos similares

3. **Reranking** reordena os resultados mais relevantes (opcional)

4. **Geração: LLM** recebe os textos + pergunta e gera a resposta

5. **Resposta final** é entregue ao usuário



COMPONENTES DA ARQUITETURA RAG

Fonte de dados:

- Reúne informações de diferentes origens: bases estruturadas, documentos, APIs e conteúdo web
- Define formatos e políticas de atualização dos dados.

Pré-processamento:

- Limpeza, normalização e extração de texto (OCR, tokenização, remoção de ruído).
- Garante consistência antes da geração dos embeddings.

Geração de embeddings

- Transforma textos em vetores numéricos por meio de modelos de linguagem de representação semântica (ex.: Sentence Transformers, OpenAI Embeddings).

Indexação e armazenamento

- Armazena os vetores em um banco de dados vetorial (ex.: FAISS, Milvus, Pinecone).
- Permite buscas rápidas por similaridade.

Recuperação

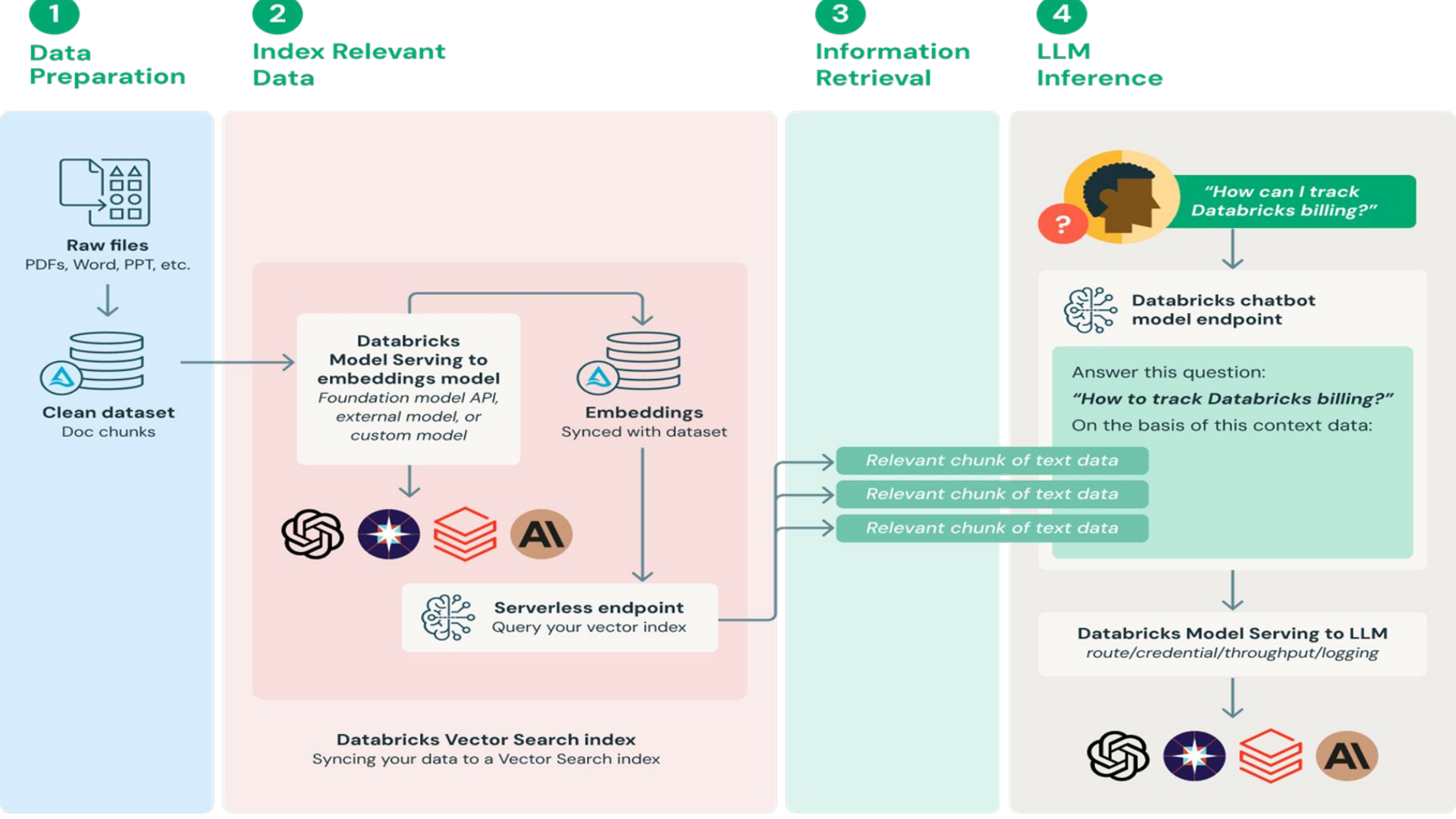
- Diante de uma consulta, realiza busca semântica e retorna os documentos mais relevantes.
- Pode incluir mecanismos de reranking para otimizar os resultados.

Geração

- O modelo generativo (ex.: LLM) recebe o contexto recuperado e produz a resposta textual coerente e contextualizada.

Pós-processamento e resposta final

- Formatar e validar a resposta final antes de exibir ao usuário.
- Pode incluir filtros, sumarização ou verificações de consistência\


1
Data Preparation
2
Index Relevant Data
3
Information Retrieval
4
LLM Inference
Raw files
PDFs, Word, PPT, etc.
Clean dataset
Doc chunks
Databricks Model Serving to embeddings model
Foundation model API, external model, or custom model
Embeddings
Synced with dataset
Serverless endpoint
Query your vector index
Databricks Vector Search index
Syncing your data to a Vector Search index
Relevant chunk of text data
Relevant chunk of text data
Relevant chunk of text data
"How can I track Databricks billing?"
Databricks chatbot model endpoint
Answer this question:
"How to track Databricks billing?"
On the basis of this context data:
Databricks Model Serving to LLM
route/credential/throughput/logging

TIPOS DE ARQUITETURA RAG

RAG Ingênuo (Naive RAG)

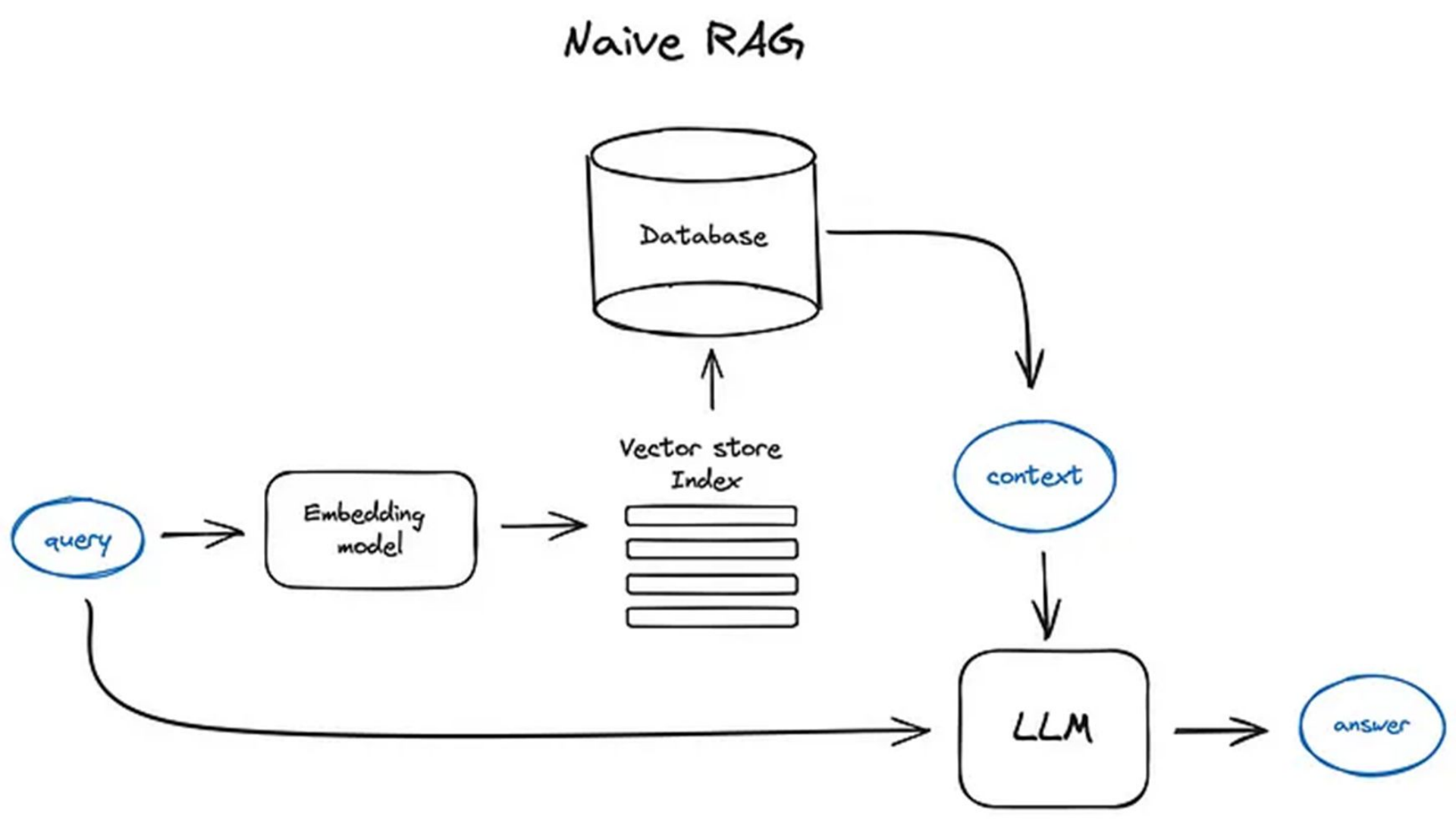
- Estrutura linear: recuperar → concatenar → gerar
- Recuperação simples (palavras-chave ou similaridade básica)
- Contexto concatenado diretamente à consulta
- Respostas baseadas apenas nos blocos recuperados (sem refinamento)

Vantagens:

- Fácil de implementar
- Boa base para prototipagem

Limitações:

- Contexto limitado
- Menor precisão em domínios complexos


Naive RAG
Database
Vector store
Index
context
query
Embedding
model
LLM
answer

TIPOS DE ARQUITETURA RAG

RAG Avançado

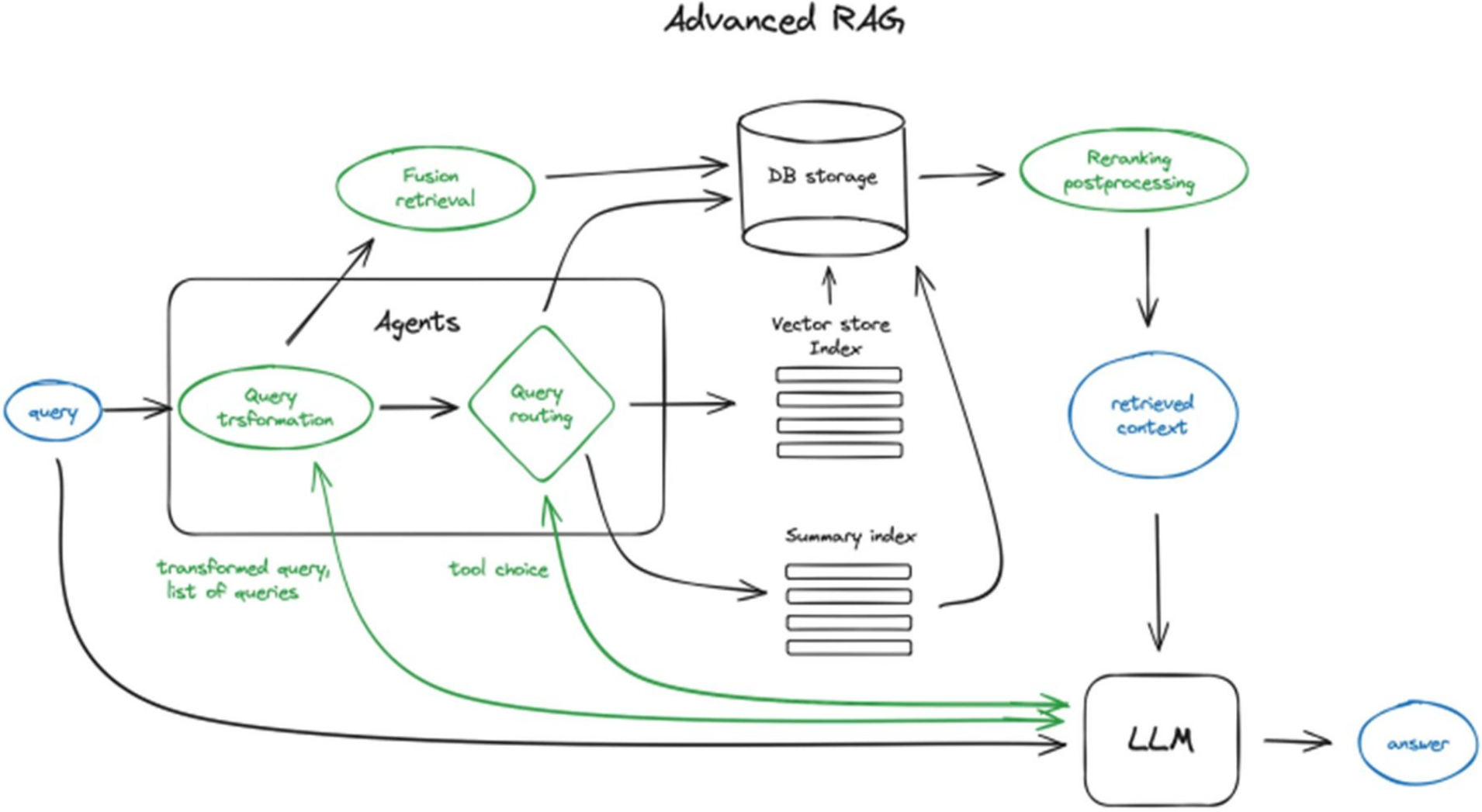
- Recuperação aprimorada com expansão de consulta e recuperação iterativa
- Uso de mecanismos de atenção para refinar o contexto
- Aplicação de pontuação de relevância e reranking
- Contexto mais preciso e dinâmico

Vantagens:

- Melhora a precisão das respostas
- Reduz redundância e ruído no contexto
- Refina dinamicamente o conteúdo com base na relevância
- Suporta tarefas mais complexas com múltiplos turnos de recuperação

Limitações:

- Maior custo computacional
- Complexidade na implementação


Advanced RAG
Fusion retrieval
DB storage
Reranking postprocessing
Agents
Vector store Index
query
Query trsformation
Query routing
retrieved context
Summary index
transformed query, list of queries
tool choice
LLM
answer

TIPOS DE ARQUITETURA RAG

RAG Modular

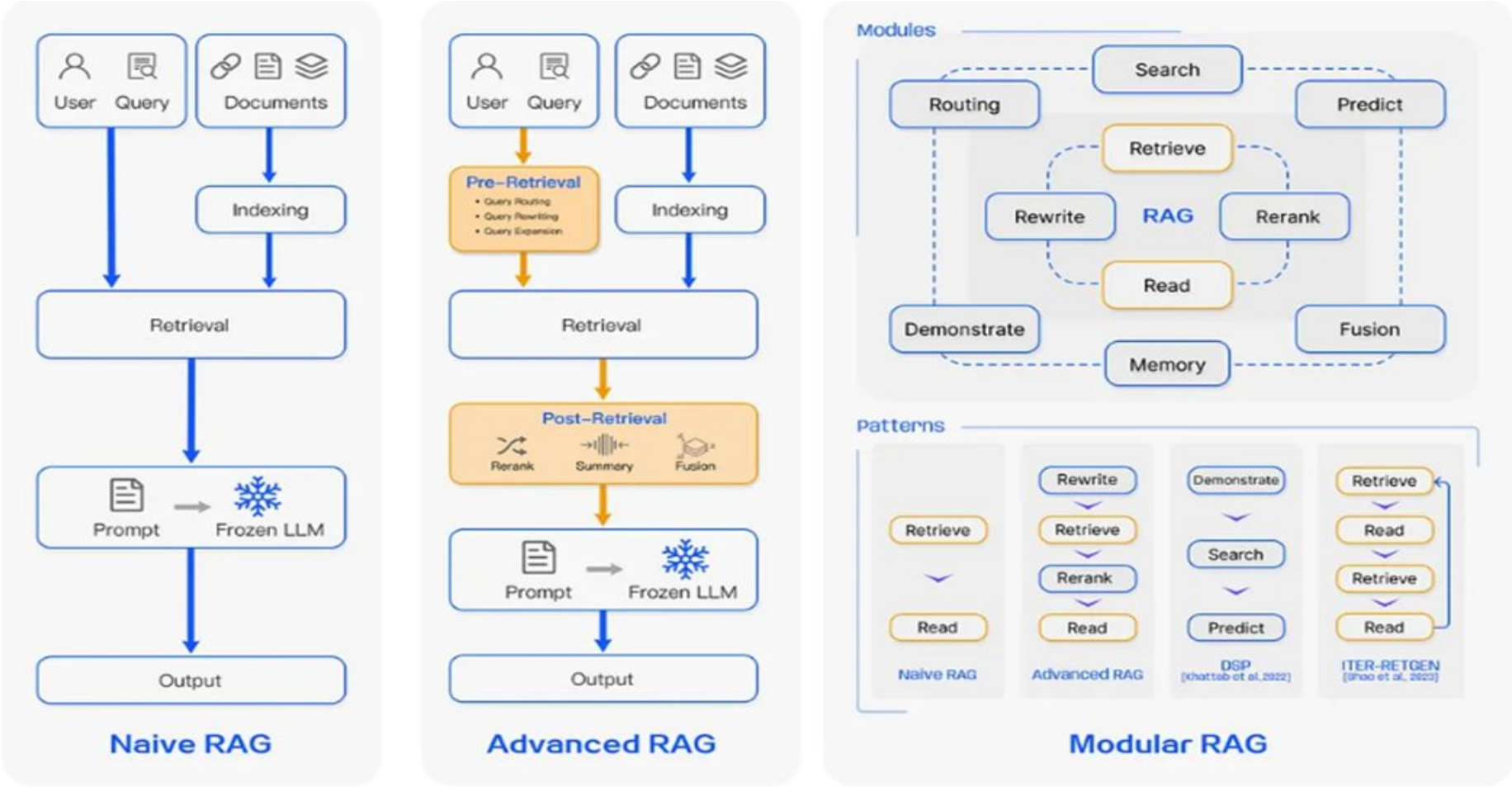
- Processo dividido em módulos independentes (recuperação, reclassificação, geração etc.)
- Permite substituir ou otimizar módulos conforme a aplicação
- Integra módulos de memória, pesquisa externa ou gráficos de conhecimento

Vantagens

- Alta flexibilidade e personalização
- Facilita experimentação e integração com outros sistemas

Limitações

- Problemas de compatibilidade entre módulos
- Aumento na latência e nos requisitos de infraestrutura
- Exige maior esforço de engenharia



Comparison between the three paradigms of RAG (Gao et al. 2024)

Técnicas Avançadas de RAG

- **Query Rewriting:** melhora a busca com reformulação de perguntas.
- **Multi-query:** divide a pergunta em várias consultas para respostas mais completas.
- **Multi-hop:** executa buscas em etapas, uma após a outra.
- **Query Routing:** direciona a busca para a fonte de dados correta.
- **Agentic RAG:** o LLM age como um agente, controlando consultas, fontes e ações.

DESAFIOS COMUNS NA IMPLEMENTAÇÃO

1. Qualidade da Recuperação

- Documentos irrelevantes reduzem a precisão das respostas
- Necessidade de selecionar bons modelos de embeddings
- Importância de ajustar métricas de similaridade e reranking
- Curadoria de dados e avaliação contínua do pipeline

2. Limitação da Janela de Contexto

- Inserção excessiva de conteúdo gera respostas truncadas ou confusas
- Estratégias de chunking devem equilibrar tamanho e coerência semântica
- Técnicas de sumarização e priorização de contexto ajudam a otimizar o uso de tokens

3. Atualização dos Dados

- Índices podem se tornar rapidamente obsoletos
- Necessário implementar pipelines de ingestão e atualização automatizados
- Dados desatualizados aumentam o risco de alucinações e imprecisões

4. Latência e Desempenho

- Recuperação em grandes volumes de dados pode gerar atrasos
- Dependência de APIs externas pode aumentar o tempo de resposta
- Requer otimização de consultas, caching e balanceamento de carga

5. Avaliação de Sistemas RAG

- Avaliação tradicional de IA não cobre aspectos híbridos (busca + geração)
- Requer métricas específicas: relevância, fundamentação e consistência
- Combinação de análise automática e revisão humana para validação da qualidade

CUSTOS E OTIMIZAÇÃO EM RAG

Custos de Operação

- **Custo por consulta**: RAG pode custar de **2x a 5x mais** que o uso direto do GPT-4 (~$0,03/1k tokens), devido a **embeddings**, **busca vetorial** e **contexto expandido**.

- **Custo-benefício**:
  - **< 1.000 consultas/dia**: Overhead pode ser aceitável pela **melhora na qualidade**.
  - **> 10.000 consultas/dia**: **Otimizações são essenciais** para manter a viabilidade econômica.

Estratégias de Otimização de Custos

- **Cache inteligente** para consultas repetidas
- **Filtragem de queries** que não exigem recuperação externa
- Uso de **modelos de embedding mais eficientes**
- **Tiers de serviço** com diferentes níveis de resposta conforme a criticidade

APLICABILIDADE

RAG na Educação

- Recupera conteúdo de **bases didáticas, artigos científicos e repositórios acadêmicos**
- Gera **respostas fundamentadas e contextualizadas** para alunos e professores
- Cria **planos de aula e materiais de estudo** com base em objetivos pedagógicos
- Facilita o acesso a **conhecimento atualizado** via busca semântica e geração guiada pelo contexto

RAG no Comércio Eletrônico

- Integra dados de **catálogos, políticas comerciais e históricos de pedidos**
- Recupera informações para **respostas personalizadas ao cliente**
- Oferece **suporte automatizado multicanal** (chat, e-mail, voz)
- **Aprimora recomendações** e **reduz custos operacionais** de atendimento

PERSPECTIVAS FUTURAS

1. Personalização Avançada

- Incorporação de perfis e histórico do usuário para geração contextualizada.
- Adaptação dinâmica das respostas com base em preferências e domínios específicos.

2. Controle de Comportamento (Customização do Modelo)

- Ajuste direto de parâmetros de geração (ex.: tom, estilo, nível técnico).
- Interfaces que permitem ao usuário guiar o raciocínio do modelo.

3. Escalabilidade e Eficiência Computacional

- Indexação distribuída e vetores comprimidos para grandes volumes de dados.
- Otimização de latência em pipelines paralelos de recuperação e geração.

PERSPECTIVAS FUTURAS

4. Modelos Híbridos e Multimodais

- Combinação de RAG com aprendizado por reforço e agentes autônomos.
- Integração de múltiplas modalidades (texto, imagem, áudio) em consultas unificadas.

5. Processamento em Tempo Real

- Recuperação e geração de respostas com latência mínima (<100 ms).
- Aplicações em assistentes inteligentes, sistemas de recomendação e monitoramento contínuo.

6. Evolução das Avaliações RAG

- Novas métricas de factual grounding e retrieval precision.
- Uso de feedback humano contínuo (RLHF) para aperfeiçoar qualidade e relevância.

Dúvidas?