Escrevendo as melhores práticas para otimizar os aplicativos RAG

AWS Recomendações

AWS Recomendações: Escrevendo as melhores práticas para otimizar os aplicativos RAG

Table of Contents

Introdução 1
Público-alvo 1
Objetivos 2
Compreendendo LLMs e RAG 3
Vetores e incorporações 5
bancos de dados vetoriais 5
Pesquisa semântica 6
Desafios nos dados de origem 7
Práticas recomendadas 9
Perguntas frequentes 11
Por que é importante otimizar documentos para aplicativos RAG? 11
Quais são alguns desafios comuns com documentos brutos que podem prejudicar o desempenho do RAG? 11
Como o uso de cabeçalhos e subtítulos pode melhorar o desempenho do RAG? 11
Por que é recomendável substituir as informações da tabela por uma sintaxe de nível simples? 11
Como a adição de resumos pode melhorar o desempenho do RAG? 12
Por que é importante definir abreviações e definir o contexto? LLMs 12
Próximas etapas e recursos 13
Recursos 13
AWS documentação 13
Outros AWS recursos 14
Histórico do documento 15
..... xvi

Escrevendo as melhores práticas para otimizar os aplicativos RAG

Ivan Cui e Samantha Stuart, Amazon Web Services

Julho de 2025 (histórico do documento)

Grandes modelos de linguagem (LLMs) revolucionaram o campo da inteligência artificial com sua notável capacidade de entender e gerar textos semelhantes aos humanos. No entanto, eles enfrentam uma limitação significativa: eles só podem trabalhar com o conhecimento contido em seus dados de treinamento. É aqui que a Retrieval Augmented Generation (RAG) ajuda. Ele oferece uma solução que combina LLMs com fontes externas de conhecimento, como dados e documentos da sua organização. Por meio de um processo de duas etapas que envolve recuperação de informações e geração de respostas, o RAG permite que os sistemas de IA acessem e incorporem up-to-date informações de várias fontes, resultando em respostas mais precisas e informadas que preenchem a lacuna entre o conhecimento do modelo estático e as necessidades dinâmicas de informações do mundo real.

Como você pode otimizar o conteúdo para recuperação em um aplicativo baseado em RAG? Este guia fornece as melhores práticas para ajudá-lo a otimizar a formatação e o estilo de escrita do conteúdo baseado em texto na base de conhecimento. A otimização do conteúdo aprimora o contexto que ajuda os aplicativos RAG a entender as informações específicas da tarefa com mais precisão. Quando o sistema recupera conteúdo altamente relevante e preciso, a qualidade da resposta do LLM melhora. A otimização do processo de entrega de contexto no nível do sistema é chamada de engenharia de contexto e é uma parte essencial das arquiteturas RAG agênticas. No RAG agente, um ou mais LLMs motivos adicionais atendam às solicitações de entrada antes da execução do RAG. Isso facilita um processo de entrega de informações em várias etapas. À medida que as arquiteturas RAG se tornam cada vez mais complexas, a otimização do conteúdo de origem continua sendo o meio mais direto de fornecer um contexto claro. LLMs Essas melhores práticas foram projetadas para ajudar você a maximizar o investimento da sua organização em um aplicativo RAG.

Público-alvo

Este guia é destinado a engenheiros de IA, cientistas de dados, engenheiros de dados ou desenvolvedores de software que estão criando aplicativos LLM com um ou mais componentes

RAG. Para entender os conceitos e recomendações deste guia, você deve estar familiarizado com os bancos de dados vetoriais e as solicitações de LLMs.

Objetivos

As recomendações deste guia podem ajudar você a alcançar o seguinte:

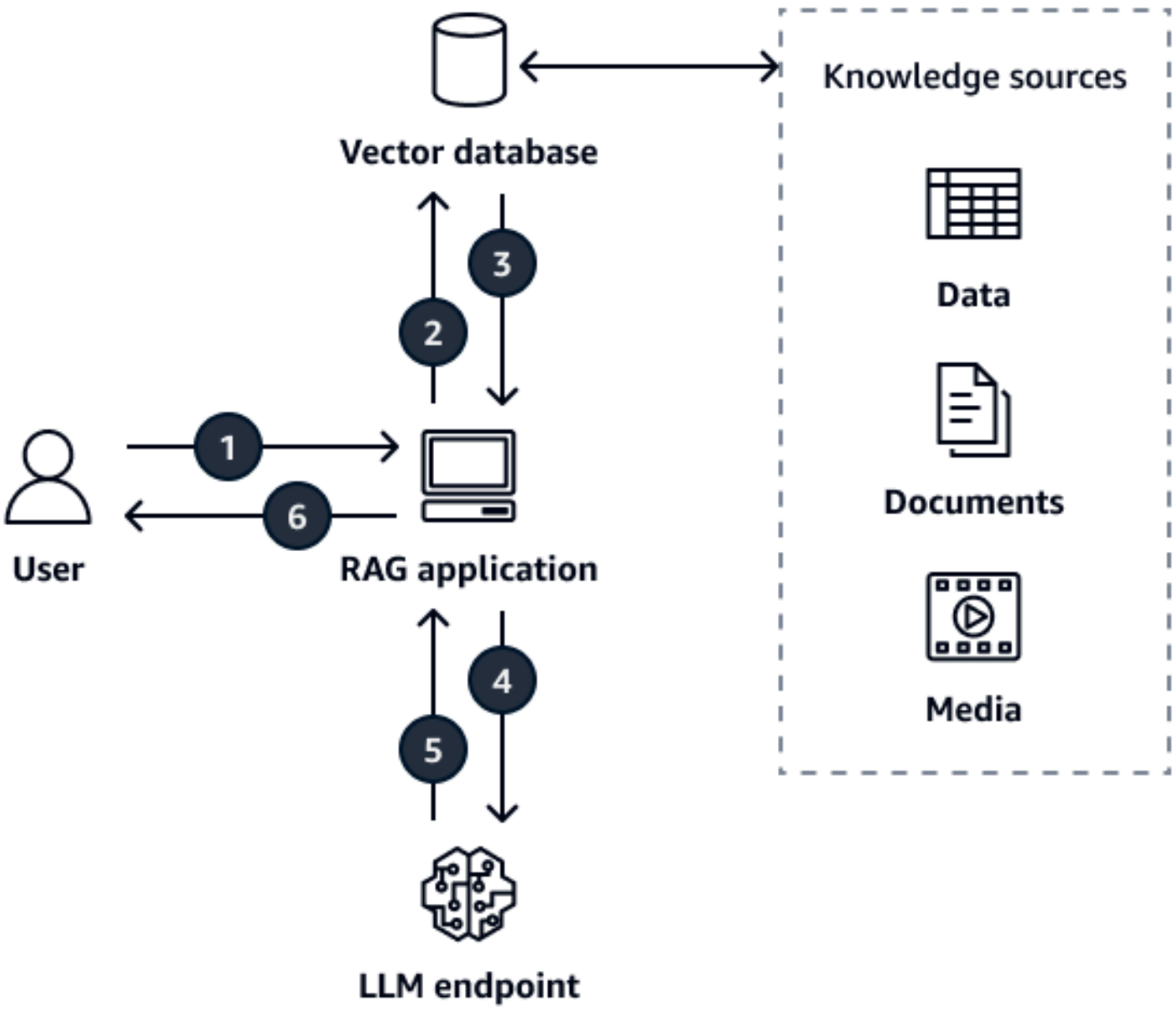
- Melhore a precisão e a relevância das respostas geradas pelos aplicativos RAG fornecendo documentos de origem bem estruturados e semanticamente ricos, otimizados para uso de tokens e redundância.
- Ajude os aplicativos do RAG a entender melhor o conhecimento e o contexto específicos do domínio, fornecendo definições e explicações claras nos documentos de origem.
- Facilite a manutenção e as atualizações da base de conhecimento para aplicativos RAG aderindo às diretrizes consistentes de formatação e estruturação em todos os documentos de origem.
- Melhore a escalabilidade das soluções RAG dividindo documentos grandes e monolíticos em unidades menores e independentes que podem ser indexadas e recuperadas com eficiência.

Compreendendo LLMs e RAG

Para entender como o aprimoramento da qualidade do documento de origem melhora a qualidade de uma resposta do RAG, você deve entender o funcionamento interno de um LLM. O verdadeiro poder dos LLMs está em sua capacidade de usar mecanismos de autoatenção e arquiteturas de transformadores. Essas técnicas avançadas permitem que os modelos processem e relacionem com eficácia diferentes partes da sequência de entrada, independentemente de sua posição ou distância dentro do texto. Esse recurso contrasta fortemente com os modelos de linguagem tradicionais, que geralmente têm dificuldades com dependências de longo alcance e compreensão do contexto. Além disso, os LLMs são treinados em uma escala sem precedentes. Alguns dos maiores modelos são compostos por trilhões de parâmetros e ingeriram terabytes de dados textuais de diversas fontes. Essa grande escala permite que os LLMs desenvolvam uma compreensão rica da linguagem, capturando nuances sutis, expressões idiomáticas e dicas contextuais que antes eram desafiadoras para os sistemas de IA. O resultado é uma classe de modelos que pode gerar texto coerente e fluente e demonstrar recursos notáveis em tarefas como resposta a perguntas, resumo de texto e até geração de código.

Para usar esses modelos, podemos recorrer a serviços como o Amazon Bedrock, que fornece acesso a uma variedade de modelos básicos da Amazon e de fornecedores terceirizados, incluindo Anthropic, Cohere e Meta. Você pode usar o Amazon Bedrock para experimentar modelos de última geração, personalizá-los e ajustá-los ou incorporá-los às suas soluções AI-powered generativas por meio de uma única API.

Embora os LLMs sejam excelentes em capturar padrões e gerar texto coerente, eles geralmente não têm acesso a informações atualizadas ou especializadas. O RAG combina o poder generativo dos LLMs com um componente de recuperação que pode acessar e incorporar informações relevantes de fontes externas, como parte do prompt materializado do LLM. Exemplos de fontes externas incluem bases de conhecimento do Amazon Bedrock, sistemas de busca inteligentes, como o Amazon Kendra, ou bancos de dados vetoriais, como o Amazon Service. OpenSearch



O diagrama descreve o seguinte fluxo de trabalho:

1. O usuário envia uma consulta ao aplicativo RAG.
2. O aplicativo RAG consulta um banco de dados vetorial que contém fontes de conhecimento, como documentos, dados ou mídia.
3. O aplicativo RAG recupera as informações relevantes do banco de dados vetoriais com base nas semelhanças semânticas entre a consulta e os documentos armazenados.
4. O aplicativo RAG aumenta o prompt original com o contexto recuperado e o envia para o endpoint do LLM.

5. O endpoint LLM gera uma resposta e a retorna ao aplicativo RAG.
6. O aplicativo RAG retorna a resposta gerada ao usuário.

Em sua essência, o RAG emprega um processo de dois estágios. No primeiro estágio, um modelo de recuperação identifica e recupera documentos ou passagens relevantes com base na consulta de entrada. Esse modelo de recuperação pode ser um sistema tradicional de recuperação de informações, um modelo de recuperação denso ou uma combinação de ambos. No segundo estágio, as informações recuperadas e a consulta original são inseridas em um LLM como um modelo de prompt totalmente materializado. Os LLMs dependem muito da qualidade do conteúdo de origem fornecido pelo componente recuperador. Eles aplicam um mecanismo de autoatenção para codificar matematicamente como o conteúdo recuperado se relaciona com a tarefa. O LLM então gera uma resposta com base na consulta e nas informações recuperadas. No RAG, controlar a qualidade dos documentos de origem recuperados representa um meio direto de melhorar a representação interna de uma tarefa por um LLM. O RAG aumenta efetivamente os dados de treinamento do LLM com dados externos relevantes. Essa abordagem permite que o RAG aproveite os pontos fortes dos LLMs e dos sistemas de recuperação, permitindo a geração de respostas mais precisas e informadas que incorporem conhecimento atual e especializado.

Vetores e incorporações

Vetores e incorporações são conceitos fundamentais em aprendizado de máquina e processamento de linguagem natural. Vetores são objetos matemáticos que representam quantidades que têm magnitude e direção. No contexto do processamento de linguagem natural (PNL), palavras, frases ou documentos são frequentemente representados como vetores em espaços vetoriais de alta dimensão. As incorporações, por outro lado, são uma forma de representar objetos como palavras ou documentos em um espaço vetorial de menor dimensão, onde as relações entre os vetores capturam semelhanças semânticas ou sintáticas. A incorporação de palavras, por exemplo, permite que palavras com significados semelhantes tenham representações vetoriais semelhantes. Isso ajuda os algoritmos a entender e processar a linguagem com mais eficiência.

bancos de dados vetoriais

Na IA generativa, um banco de dados vetorial é um banco de dados que armazena e gerencia representações vetoriais de documentos, consultas ou outros objetos. Ele foi projetado para armazenar e recuperar vetores com eficiência. Isso oferece suporte a operações rápidas e escaláveis, como pesquisa semântica e correspondência por similaridade. Os bancos de dados

vetoriais indexam vetores usando estruturas de dados especializadas, como gráficos Hierarchical Navigable Small World (HNSW) ou algoritmos de Neighbors (KNN). K-Nearest Essas estruturas de dados permitem pesquisas rápidas nos vizinhos mais próximos, possibilitando encontrar rapidamente vetores semelhantes no banco de dados.

Pesquisa semântica

A pesquisa semântica é uma técnica que melhora a relevância dos resultados da pesquisa ao entender a intenção e o contexto da consulta, em vez de apenas combinar palavras-chave. Em termos técnicos, a pesquisa semântica envolve a comparação das representações vetoriais da consulta e dos documentos no banco de dados para encontrar as correspondências mais relevantes. Diferentes estratégias de recuperação podem ser usadas para pesquisa semântica, incluindo, mas não se limitando a:

- HNSW — Uma estrutura de dados baseada em gráficos que organiza vetores de forma a tornar eficiente a busca por vizinhos mais próximos.
- KNN — Um algoritmo que encontra os K vetores mais próximos de um vetor de consulta com base em uma métrica de distância, como similaridade de cosseno.
- Similaridade de cosseno — Uma medida de similaridade entre dois vetores diferentes de zero que mede o cosseno do ângulo entre eles. É frequentemente usado na pesquisa semântica para comparar a direção dos vetores em um espaço de alta dimensão.
- Locality-sensitive hashing (LSH) — Uma técnica que faz o hash de vetores semelhantes aos mesmos buckets ou a buckets próximos com alta probabilidade. Isso permite pesquisas aproximadas no vizinho mais próximo, que podem ser mais rápidas do que pesquisas exatas em espaços de alta dimensão.

Desafios nos dados de origem que afetam os aplicativos RAG

Um dos desafios significativos no desenvolvimento de um aplicativo ideal de geração aumentada de recuperação (RAG) está na natureza dos dados ou documentos brutos usados. Muitas vezes, as empresas usam documentos existentes que foram criados para referência humana. Esses documentos geralmente incluem hiperlinks e capturas de tela de imagens para promover a compreensão. No entanto, esses elementos obstruem a recuperação semântica devido aos limites do token do trecho. Isso resulta em baixo desempenho do recuperador.

A seguir estão os desafios mais comuns de documentos brutos para um aplicativo RAG ideal:

- Falta de formatação e metadados estruturados — documentos brutos podem não ter cabeçalhos de seção, subtítulos ou metadados claros. Isso torna difícil identificar e extrair informações relevantes. Por exemplo, um documento longo sem cabeçalhos claros pode dificultar a determinação do contexto de informações específicas.
- Linguagem informal e inconsistente — Documentos brutos geralmente contêm linguagem informal ou terminologia inconsistente. Isso pode confundir os modelos RAG. Por exemplo, abreviações que não estão definidas no documento ou que já são conhecidas pelo LLM podem ser usadas em todo o documento.
- Verbosidade e redundância — Documentos brutos podem ser detalhados e conter informações desnecessárias ou redundantes. Isso pode sobrecarregar os modelos RAG, levando a respostas menos concisas e relevantes. Os exemplos incluem um documento que repete as mesmas informações várias vezes ou vários documentos que contêm informações semelhantes ou contraditórias.
- Termos e frases ambíguos — documentos brutos podem conter termos ou frases ambíguos que podem ser interpretados de várias maneiras. Essa ambigüidade pode levar a interpretações errôneas pelos modelos RAG e respostas imprecisas. Por exemplo, um documento que usa um termo com vários significados pode resultar em uma resposta que não se alinha ao significado pretendido.
- Injeção de elementos gráficos e de hiperlink — Documentos brutos que contêm gráficos e informações de hiperlinks funcionam bem para consumo humano. No entanto, esses elementos podem consumir o limite do token de recuperação. O resultado é que os trechos podem estar incompletos. Por exemplo, gráficos e hiperlinks URLs são retornados como parte da

recuperação, que usa os tokens de recuperação, e faltam as principais informações dos parágrafos subsequentes.

- Falta de conhecimento ou contexto específico do domínio — Os documentos brutos podem não ter o conhecimento específico do domínio ou o contexto necessários para uma geração precisa. Isso pode limitar a capacidade dos modelos RAG de gerar respostas relevantes e precisas. Um exemplo é um documento que faz referência a conceitos especializados sem fornecer contexto. Isso pode levar a respostas que não são significativas em um determinado domínio.

Embora essa lista não seja abrangente, ela fornece um ponto de partida para as empresas pensarem sobre o que não está funcionando e por quê. Os documentos podem ter um ou mais desses desafios. A chave para otimizar um aplicativo RAG é usar um conjunto de documentos que sigam as melhores práticas de redação que otimizem a recuperação.

Práticas recomendadas de documentação para aplicativos RAG

O desenvolvimento de um aplicativo bem-sucedido de Retrieval-Augmented Generation (RAG) requer uma análise cuidadosa de vários fatores relacionados a documentos para otimizar seu desempenho. As melhores práticas nesta seção são selecionadas com base na experiência na criação de sistemas RAG com muitos líderes organizacionais. A seguir estão algumas das principais práticas recomendadas para documentos para aprimorar a eficácia de seu aplicativo RAG:

- Use cabeçalhos e subtítulos adequadamente — Organizar seu conteúdo com cabeçalhos e subtítulos claros melhora a legibilidade e ajuda os modelos do RAG a entender a estrutura de seus documentos. Essa prática permite que os modelos naveguem melhor e extraiam informações dos documentos, o que melhora a qualidade das respostas geradas.
- Certifique-se de que a numeração seja sequencial — Ao usar listas numeradas, é importante manter a numeração correta para evitar confusão. Certifique-se de que cada item da lista seja numerado sequencialmente sem pular números. Isso ajuda a manter a clareza e a coerência em seu conteúdo.
- Adicionar transições entre itens da lista — Fornecer transições entre itens em uma lista numerada ou com marcadores ajuda a orientar o LLM pelo conteúdo. Por exemplo, você pode usar frases como “Depois de concluir a etapa 2, faça...” para conectar ideias e melhorar o fluxo de informações.
- Substituir tabelas — Evite usar tabelas. Formate essas informações em listas com marcadores de vários níveis ou em uma sintaxe de nível simples. A sintaxe de nível plano é organizar elementos ou itens no mesmo nível hierárquico, sem níveis aninhados de subordinação. Essas estruturas ajudam LLMs a digerir as informações. Como a maioria dos documentos indexados é lida da esquerda para a direita, a sintaxe de nível simples permite que as informações sejam acompanhadas de forma mais coerente sem precisar referenciar uma dimensão adicional. Esse formato é mais propício para aplicativos RAG porque apresenta as informações de forma estruturada e de fácil digestão.
- Pré-processe informações gráficas para maior eficiência — o multimodal LLMs pode ingerir imagens e texto. Reduza a resolução das imagens, remova imagens redundantes e descreva o conteúdo dos elementos gráficos em formato de texto. Essas medidas melhoram o contexto significativo, evitam o consumo desnecessário de tokens e melhoram a acessibilidade dos modelos RAG.

- Adicione iniciadores de sessão para consultas comuns — Ao abordar questões ou tarefas comuns, como “Como faço para solicitar software?” , adicione um iniciador de sessão que faça a transição do leitor para o processo. Por exemplo, você pode adicionar “Se você deseja solicitar um software, siga as etapas abaixo...”. Isso ajuda a criar uma alta correspondência semântica, o que ajuda o LLM a construir uma resposta coesa.
- Adicione um resumo a cada seção — Depois de cada título ou subtítulo, adicione um resumo breve e conciso do conteúdo dessa seção. Isso pode aumentar a cobertura semântica e reforçar os pontos-chave. Isso melhora a precisão da pesquisa por similaridade no espaço de incorporação, melhorando assim o desempenho do aplicativo RAG. Isso é particularmente útil se o documento for destinado tanto ao LLM quanto ao consumo humano ou se elementos gráficos e de tabela forem necessários.
- Desambiguação — Os documentos devem ser concisos e focados. LLMs gere respostas com base em trechos recuperados, para que a desambiguação ajude o modelo a usar informações claras e relevantes. Isso resulta em respostas mais precisas e informativas.
- Defina abreviações e defina o contexto — LLMs são treinados em grande quantidade de dados da Internet e, na maioria das vezes, não têm o contexto dos documentos internos de uma empresa. Portanto, definir o contexto, definir abreviações e evitar ou definir a terminologia específica da empresa ajuda o LLM a entender seus dados corporativos. Isso ajuda o LLM a responder às perguntas com mais precisão e pode ajudar a evitar alucinações.
- Reestruture documentos grandes em documentos menores para marcação e indexação eficientes — Evite indexar um documento grande que contenha vários subtópicos. Considere dividir o documento grande em documentos menores e independentes que tenham títulos claros. Isso melhora a indexação e a marcação.

Perguntas frequentes

Por que é importante otimizar documentos para aplicativos RAG?

Os documentos brutos geralmente são escritos para consumo humano sem considerar os requisitos de sistemas avançados de IA, como aplicativos de geração aumentada de recuperação (RAG). A otimização de documentos seguindo as melhores práticas pode melhorar significativamente o desempenho e a precisão dos aplicativos RAG, fornecendo informações estruturadas, inequívocas e relevantes aos modelos.

Quais são alguns desafios comuns com documentos brutos que podem prejudicar o desempenho do RAG?

Alguns dos principais desafios incluem falta de formatação e metadados estruturados, linguagem informal ou inconsistente, verbosidade e redundância, termos e frases ambíguos, inclusão de elementos de hiperlink e falta de contexto específico do domínio. Esses problemas podem confundir os modelos RAG e levar a respostas imprecisas ou irrelevantes. Para obter mais informações, consulte Desafios nos dados de origem que afetam os aplicativos RAG neste guia.

Como o uso de cabeçalhos e subtítulos pode melhorar o desempenho do RAG?

Cabeçalhos e subtítulos claros ajudam os modelos do RAG a entender a estrutura e o contexto do conteúdo. Isso permite que eles naveguem melhor e extraiam informações relevantes dos documentos e melhoram a qualidade das respostas geradas. Para obter mais informações, consulte as melhores práticas de documentação para aplicativos RAG neste guia.

Por que é recomendável substituir as informações da tabela por uma sintaxe de nível simples?

Pode ser difícil para os modelos RAG interpretar tabelas porque eles exigem uma compreensão da estrutura bidimensional. Apresentar as informações da tabela em uma sintaxe de nível simples ou em uma lista com marcadores ajuda os modelos a processar as informações com mais facilidade,

levando a um melhor desempenho. Para obter mais informações, consulte as melhores práticas de documentação para aplicativos RAG neste guia.

Como a adição de resumos pode melhorar o desempenho do RAG?

Incluir resumos concisos no início de cada seção ou subseção pode aumentar a cobertura semântica e reforçar os pontos-chave. Isso melhora a precisão das pesquisas por similaridade no espaço de incorporação, o que, em última análise, melhora o desempenho do aplicativo RAG. Para obter mais informações, consulte as melhores práticas de documentação para aplicativos RAG neste guia.

Por que é importante definir abreviações e definir o contexto? LLMs

LLMs são treinados em uma ampla variedade de dados, mas não têm contexto para abreviações ou terminologias específicas da empresa. Definir abreviações e fornecer contexto ajuda a LLMs entender e responder com mais precisão. Isso pode ajudar a evitar alucinações ou interpretações errôneas. Para obter mais informações, consulte as melhores práticas de documentação para aplicativos RAG neste guia.

Próximas etapas e recursos

Este guia discute um conjunto de desafios em nível de documento para aplicativos RAG e as melhores práticas para mitigá-los. Esses aprendizados são organizados a partir de entrevistas e discussões com líderes do setor e são apoiados por casos de uso corporativo.

Para começar a otimizar seus documentos para aplicativos RAG, recomendamos realizar uma auditoria de seus documentos existentes. Identifique as áreas que representam desafios para a aplicação do RAG. Os exemplos incluem falta de estrutura, linguagem ambígua ou uso excessivo de elementos gráficos. Priorize documentos que são acessados com frequência ou que são essenciais para suas operações comerciais. Colabore com especialistas no assunto para implementar as melhores práticas deste guia. Certifique-se de que os documentos sejam reestruturados com cabeçalhos claros, linguagem concisa e elementos de configuração de contexto. Para novos documentos, estabeleça diretrizes e modelos que garantam a consistência e ajudem os autores a aderir às melhores práticas. Além disso, considere investir em ferramentas ou serviços que possam automatizar aspectos do processo de otimização de documentos, como o uso de IA generativa para reestruturar documentos. Ao adotar uma abordagem proativa para a otimização de documentos, você pode liberar todo o potencial dos aplicativos RAG e gerar resultados mais precisos e esclarecedores em toda a sua organização.

Os recursos a seguir podem ajudá-lo a entender e criar aplicativos RAG em sua organização.

Recursos

AWS documentação

- Escolha de um banco de dados AWS vetoriais para casos de uso do RAG (orientação AWS prescritiva)
- Implemente um caso de uso do RAG AWS usando o Terraform e o Amazon Bedrock (AWS orientação prescritiva)
- Desenvolva assistentes avançados baseados em bate-papo com IA generativa usando RAG e ReAct prompting (orientação prescritiva)AWS
- Recupere dados e gere respostas de IA com as bases de conhecimento do Amazon Bedrock (documentação do Amazon Bedrock)
- Geração aumentada de recuperação (documentação da Amazon SageMaker AI)

- Opções e arquiteturas de geração aumentada de recuperação ativadas AWS(AWS orientação prescritiva)

Outros AWS recursos

- Crie um assistente multimodal com RAG avançado e Amazon Bedrock (postagem no blog)AWS

Histórico do documento

A tabela a seguir descreve alterações significativas feitas neste guia. Se desejar receber notificações sobre futuras atualizações, inscreva-se em um feed RSS.

Alteração	Descrição	Data
Publicação inicial	—	24 de julho de 2025

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.