00:03-24:00

Building Reliable and Accurate Retrieval-Augmented Generation (RAG) Applications

Introduction

Knowledge management is a crucial application for AI due to the vast amount of unstructured data in organizations. RAG (Retrieval-Augmented Generation) enables large language models (LLMs) to access private knowledge by retrieving relevant information from a database. This article explores techniques to enhance the reliability and accuracy of RAG applications.

Best Practices for RAG Applications

1. Better Data Parsing

Complex data formats like PDF and PowerPoint can hinder data extraction for LLMs. Consider using tools like Llama Parts for PDF processing and Fire Crawl for website scraping.

2. Optimizing Chunk Size

Chunking documents into smaller units for Vector search is crucial. Experiment with different chunk sizes to determine the optimal balance between context and performance.

3. Re-ranking

To improve retrieval accuracy, use a ranker to prioritize the most relevant chunks from a Vector search. This ensures that the LLM focuses on the most important information.

4. Hybrid Search

In cases where Vector search is not optimal, consider using hybrid search to combine Vector and keyword search results.

Agentic RAG

Agentic RAG enables dynamic question analysis and optimization.

1. Query Translation and Planning

Modify user queries to improve retrieval friendliness using agents. Abstract queries to retrieve more comprehensive results.

2. Metadata Filtering and Routing

Utilize metadata like title, year, and country to filter database searches, delivering more relevant results.

3. Corrective RAG Agents

Incorporate self-reflection into RAG pipelines. Have agents evaluate retrieved documents and initiate additional internet searches to ensure accuracy.

Example: Corrective RAG Agent with Llama Studio

Build a corrective RAG agent on a local machine using Llama Studio for data parsing and Fire Crawl for website crawling. Graph Language (GL) is used for agent creation.