M08a - RAG - Prozess

Stand: 03.2025

Ein RAG-System erweitert ein Large Language Model (LLM) durch den Zugriff auf externe Wissensquellen. Dies ermöglicht es, spezifische und aktuelle Informationen aus externen Dokumenten bereitzustellen, die nicht direkt im LLM enthalten sind. In diesem Kontext werden Datenschutz- und Datensicherheitsdokumente im PDF-Format über LangChain verarbeitet, um eine fundierte und faktenbasierte Generierung von Antworten zu gewährleisten.

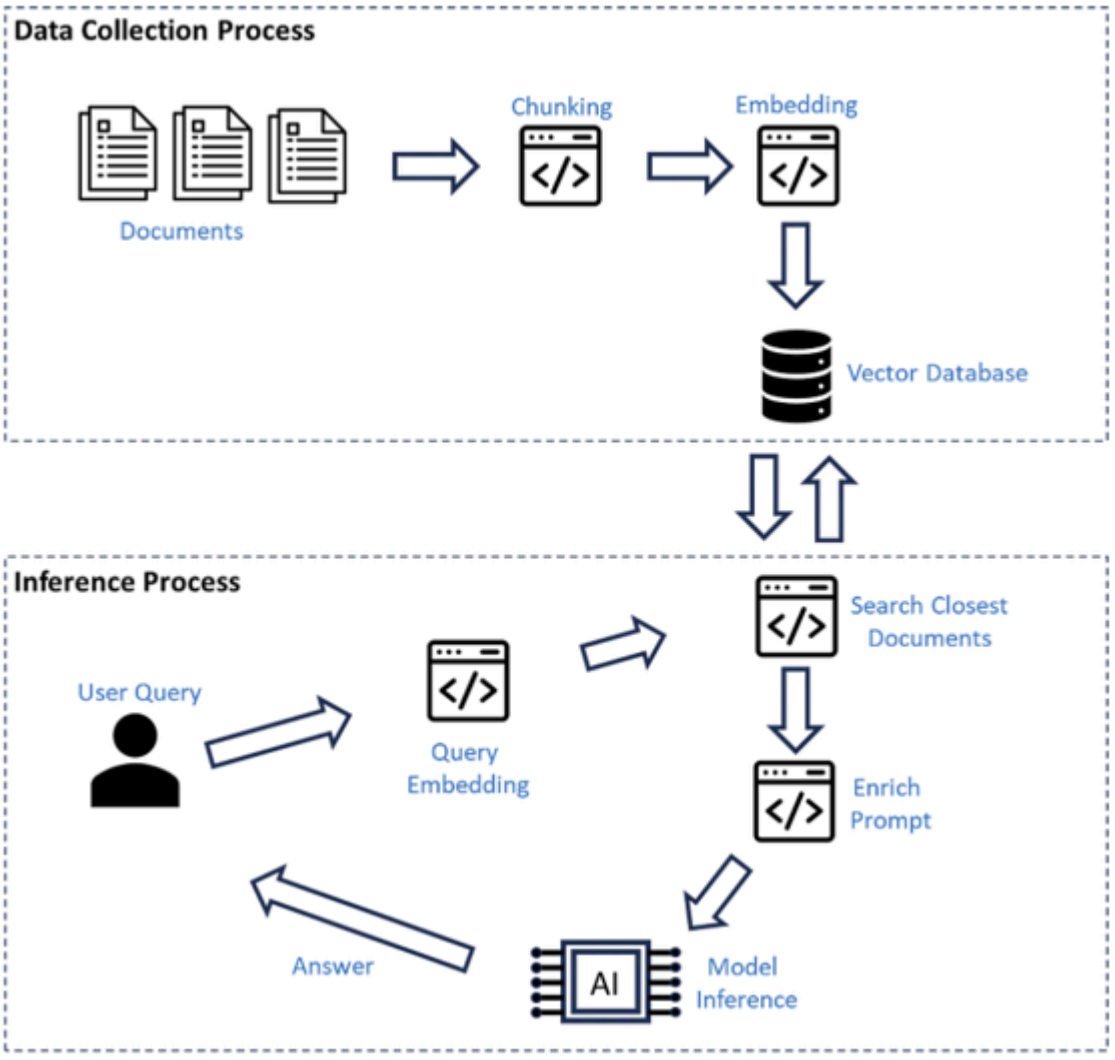


Bild: 4 Strategies to Optimize Retrieval-Augmented Generation (RAG) | by Carlos Jose Garces Rodrigues | AI Advances

1 Textvorverarbeitung

Damit die Dokumente effizient durchsuchbar und für das LLM nutzbar gemacht werden, erfolgt eine mehrstufige Vorverarbeitung:

1. **Textnormalisierung:** Sonderzeichen werden entfernt, Zeichensetzung und Groß-/Kleinschreibung harmonisiert sowie irrelevante Informationen herausgefiltert. Dies reduziert die Komplexität und verbessert die Qualität der späteren Embeddings.
2. **Tokenisierung:** Der Text wird in Tokens zerlegt, die jeweils eine eindeutige ID erhalten. Dies ist notwendig, um die Texte numerisch weiterverarbeiten zu können.
3. **Chunking:** Die Dokumente werden in kleinere Segmente (Chunks) unterteilt, um den Kontext möglichst zu erhalten. Der **Recursive Character Text Splitter** z.B. stellt sicher, dass die Chunks eine sinnvolle Länge aufweisen, um relevante Inhalte effizient abrufbar zu machen.

2 Embeddings

Um semantische Ähnlichkeiten zwischen Textpassagen zu berechnen, werden die Chunks in numerische Vektoren umgewandelt. Hierfür wird ein **Embedding-Modell** wie `text-embedding-ada-002` oder `sentence-transformers` genutzt. Diese Embeddings werden in einer **Vektordatenbank** (z. B. FAISS) gespeichert und können mittels Ähnlichkeitsberechnungen effizient durchsucht werden.

Optimierungsmöglichkeiten:

- **Hybrid-Suche:** Kombination dichten und spärlichen Vektorrepräsentationen für präzisere Abrufresultate.
- **Dynamische Chunk-Anpassung:** Optimierung der Segmentgrößen basierend auf Textstruktur und Relevanzbewertung.

3 Retrieval und Prompting

Bei einer Benutzeranfrage wird der eingegebene Text ebenfalls in einen Vektor umgewandelt und mit den gespeicherten Embeddings verglichen. Die relevantesten Chunks werden abgerufen und als Kontext an das LLM übergeben. Die Qualität der Antwort hängt entscheidend von der **Prompt-Strukturierung** ab.

Verbesserungsstrategien:

- **Query Expansion:** Ergänzung der Suchanfrage um Synonyme oder verwandte Begriffe zur Verbesserung der Trefferquote.
- **Re-Ranking:** Gewichtung und Neuanordnung der abgerufenen Chunks nach Relevanz, um nur die relevantesten Informationen an das LLM weiterzugeben.

4 Generierung der Antwort durch das LLM

Das LLM verarbeitet die übergebenen Informationen und generiert eine fundierte Antwort. Die **Temperatur-Einstellung** bestimmt dabei die Kreativität der Antwort:

- Niedrige Werte (z. B. 0.2) → faktenbasierte, deterministische Antworten.
- Höhere Werte (z. B. 0.8) → kreativere und diversere Antworten.

Erweiterte Techniken wie **Chain-of-Thought-Prompting** können die Argumentationsführung des LLM verbessern, indem die Antwort schrittweise hergeleitet wird.

5 Komponenten der RAG-Pipeline

Ein RAG-System besteht aus mehreren miteinander verknüpften Modulen:

- **Retrieval-Modul:** Sucht relevante Textpassagen aus der Wissensquelle.
- **Prompting-Modul:** Strukturiert den Kontext für das LLM.
- **Generierungs-Modul:** Erstellt die endgültige Antwort basierend auf den abgerufenen Informationen.
- **Feedback-Modul:** Nutzerbewertungen zur Verbesserung der Antwortqualität und zur iterativen Optimierung der Pipeline.

6 Evaluierung und Optimierung

Die Qualität eines RAG-Systems wird anhand von zwei Hauptmetriken bewertet:

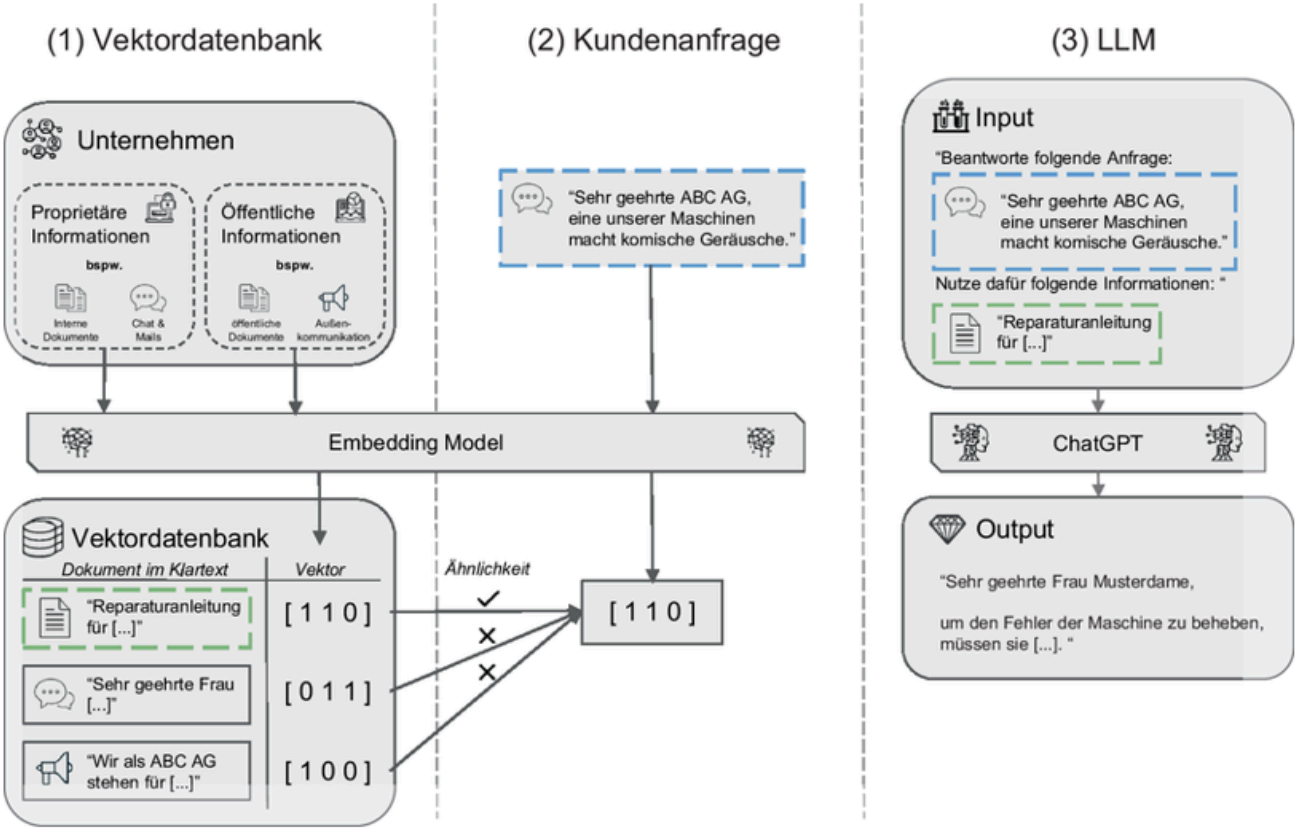
1. **Retrieval-Qualität:** Präzision und Vollständigkeit der abgerufenen Textsegmente.
2. **Generierungsqualität:** Korrektheit und Relevanz der durch das LLM erstellten Antworten.

Zusätzliche Optimierungen können durch **A/B-Tests mit verschiedenen Embedding-Modellen**, Anpassung der Chunk-Größen und Kombination verschiedener Retrieval-Strategien erreicht werden.

📝 Fazit

Ein RAG-System bietet erhebliche Vorteile für die Generierung fundierter, aktueller Antworten durch die Integration externer Wissensquellen. Die Qualität hängt von der Wahl der Embedding-Modelle, der Retrieval-Methoden und der Prompt-Strategien ab. Durch kontinuierliche Optimierung und Evaluierung lässt sich ein effizientes und präzises System für den jeweiligen Anwendungsfall entwickeln.

7 Praxisbeispiel



Quelle: [Buxmann et al.: Die Nutzung von ChatGPT in Unternehmen, 2024]