

RAG-Bericht

Mauro Schegg^{a,1}

^aOST – Ostschweizer Fachhochschule

1. Systemübersicht

Dieses Retrieval-Augmented-Generation (RAG)-System ist ein Chatbot, der wie in Abbildung 1 dargestellt aufgebaut ist. Als Ausgangsbasis diente die Aufgabenstellung dieses Projekts.

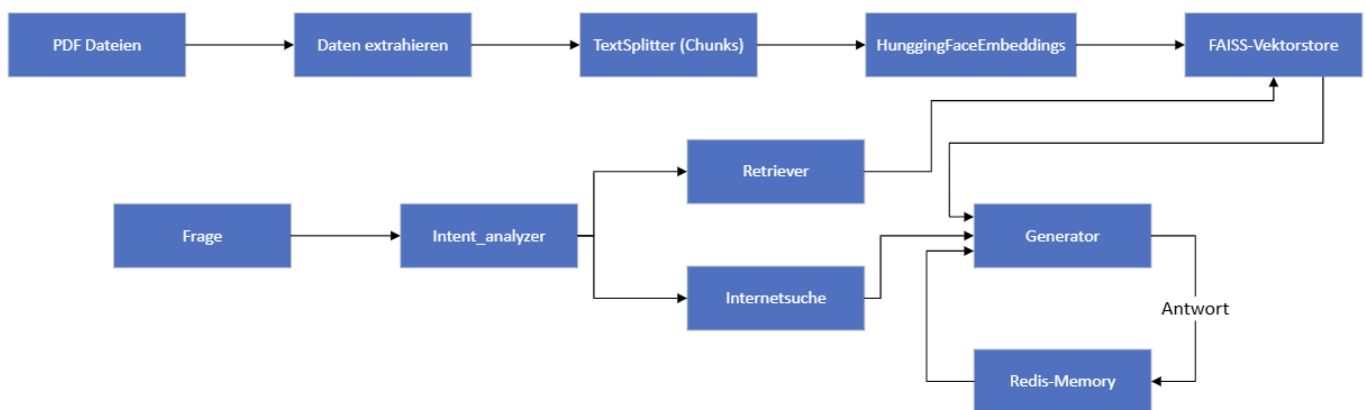


Abbildung 1. Architektur des RAG-Systems.

Das System wurde auf die Verarbeitung von PDF-Dateien ausgelegt. Der Textinhalt wird zunächst extrahiert und mit einem *RecursiveCharacterTextSplitter* in Chunks unterteilt. Diese Chunks werden mithilfe von *HuggingFaceEmbeddings* in Vektoren umgewandelt und in einer *FAISS*-Vektordatenbank gespeichert.

Die Orchestrierung der Agenten erfolgt über *LangGraph*. Bei einer Nutzeranfrage analysiert ein Klassifikator das Ziel:

Frage beantworten, Zusammenfassen oder **Internetsuche**.

Je nach Absicht wird ein spezialisierter Agent aktiviert:

- Für Fragen oder Zusammenfassungen wird der *RetrieverAgent* verwendet, um relevante Informationen aus der Vektordatenbank abzurufen.
- Für Internetrecherchen wird über *DuckDuckGo* gesucht.

Die Informationen werden anschließend an den *GeneratorAgent* übergeben, welcher unter Verwendung des in *Redis* gespeicherten Chatverlaufs eine Antwort generiert.

2. Benutzeroberfläche in Streamlit

Die Benutzeroberfläche wurde mit Streamlit umgesetzt. Die folgenden Abbildungen zeigen den Ablauf:



Abbildung 2. Startfenster der Streamlit-Anwendung.

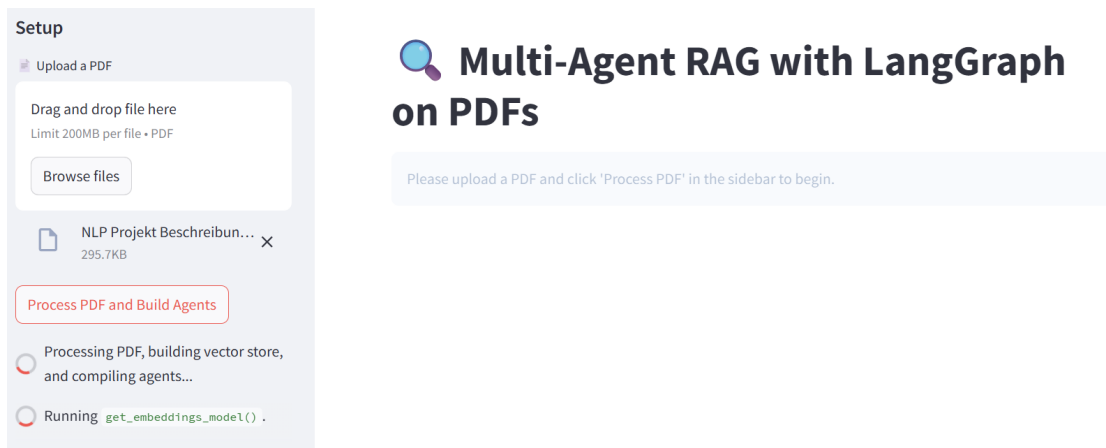


Abbildung 3. Nach dem Hochladen eines PDFs: Verarbeitung und Agenteninitialisierung.

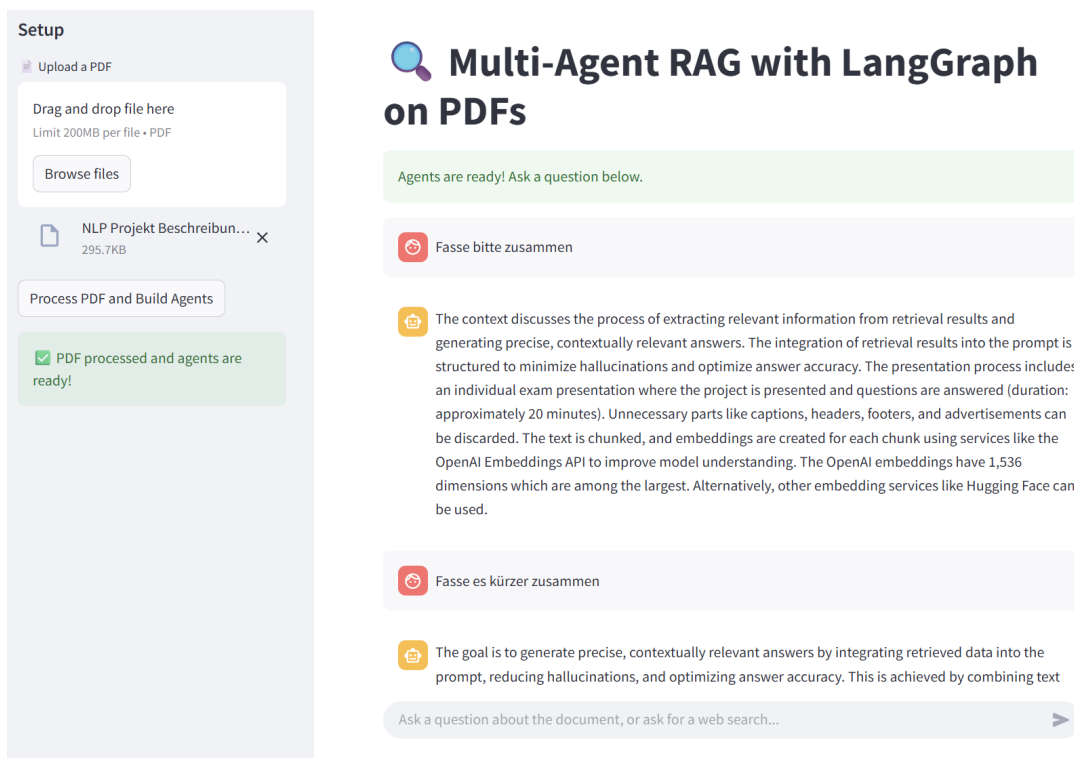



Abbildung 4. Chatfenster zur Eingabe von Fragen und Anzeige der Antworten.

3. Funktionsnachweise

Die Funktionalität wurde durch gezielte Tests überprüft:

3.1. Zusammenfassung

Die erste getestete Funktion war die automatische Zusammenfassung eines Dokuments. Das Ergebnis ist in Abbildung 5 dargestellt und wirkt inhaltlich korrekt.

 Fasse bitte zusammen



 The context discusses the process of extracting relevant information from retrieval results and generating precise, contextually relevant answers. The integration of retrieval results into the prompt is structured to minimize hallucinations and optimize answer accuracy. The presentation process includes an individual exam presentation where the project is presented and questions are answered (duration: approximately 20 minutes). Unnecessary parts like captions, headers, footers, and advertisements can be discarded. The text is chunked, and embeddings are created for each chunk using services like the OpenAI Embeddings API to improve model understanding. The OpenAI embeddings have 1,536 dimensions which are among the largest. Alternatively, other embedding services like Hugging Face can be used.

Abbildung 5. Automatisch generierte Zusammenfassung.

Anschließend wurde das System aufgefordert, eine ausführlichere Zusammenfassung zu erstellen. Die erweiterte Antwort ist in Abbildung 6 dargestellt und zeigt, dass das System auf Feedback reagieren kann.

 mache deine Antwort länger

 In the project NLP for the spring semester 2024/2025, Shao Jü Woo is focusing on the architecture of traditional Retrieval-and-Generation (RAG) based chatbots. The architecture of a semantic search application usually consists of a component for data ingestion and transformation, along with a semantic search component that not only generates an answer to a user's query (user question) but also serves as the foundation for the chatbot architecture.

The main components are:

1. Data Ingestion & Transformation: Data files containing text need to be extracted, cleaned, and converted into a format and output structure that an Language Model (LLM) can understand (like JSON). This component generally provides better performance when it is possible to isolate crucial text data and discard irrelevant data (such as image captions, headers/footers, and advertisements).

The following steps are included in the process of generating longer answers:

4. Retrieval of the actual texts of the discovered documents
5. Creation of a new prompt that encompasses both the user's question and the context of the found document
6. Handover of the newly created prompt to the Speech Model
7. Return of the response to the user

There are also other ways to enhance the capabilities of chatbots:

- Use of special Chat History or Memory functions
- Utilization of frameworks like LangGraph that can work with any LLM including Open-Source models as long as they support Tool-Calls


Moreover, Workflows are modeled as nodes (agents, tools) and edges (data flows), enabling loops, parallel processing, and dynamic decision-making. For instance, a RetrieverAgent retrieves documents while a GeneratorAgent creates responses, with feedback loops for improvement.

Regarding data storage, Vector Databases like FAISS, Qdrant, or Milvus could be considered as vector databases.

Abbildung 6. Längere Zusammenfassung auf Nutzerwunsch.

3.2. Internetsuche

Die Fähigkeit zur Internetsuche wurde in Abbildung 7 getestet. Auch diese Funktion lieferte korrekte und sinnvolle Ergebnisse.

 suche Online zu RAG


 Based on the provided web search results, RAG (Retrieval-Augmented Generation) is a technology that helps make internal knowledge of a company more accessible by enabling conversations with documents using Large Language Models (LLMs). RAG also allows developers to supply the LLM with the latest research findings, statistics, or news from social media, news sites, or other frequently updated information sources. These chunks are turned into searchable "building blocks" that enable easy and quick retrieval of accurate information. When the data preparation is complete, a search system (Retrieval System) is established to specifically search these prepared chunks for relevant information. Therefore, if you're looking for more information online about RAG, you may want to search for topics like "RAG technology", "RAG for enterprise knowledge access", or "RAG and LLMs". You could also explore specific use cases or companies that employ RAG technology.

Abbildung 7. Beispiel für eine erfolgreiche Internetsuche.

3.3. Prüfung auf Halluzinationen

Schließlich wurde das System auf Halluzinationen getestet. Dazu wurde eine Frage zu "Brennholz" gestellt, ein Begriff, der im PDF nicht vorkommt. Die korrekte Antwort war, dass keine entsprechenden Informationen im Text gefunden wurden (siehe Abbildung 8).

Was steht im Dokument zu Brennholz?

Agent Workflow Steps

- Agent: Intent Analyzer - Analyzing question...
- Intent Detected: `question_answering`
- Data Source Hint: `tech_docs`
- Router - Deciding next step...
- Route -> Document Retrieval
- Agent: Retriever - Searching PDF documents...
- Agent: Generator - Formulating the answer...

Based on the context provided, the document does not contain information specifically about "Brennholz" (German for firewood). The text focuses on the use of Language Models (LLMs) and their capabilities in handling complex and domain-specific queries by integrating relevant documents into the model using document embeddings. Therefore, the context does not provide an answer to your question.

Abbildung 8. Test auf Halluzination: Keine falsche Antwort gegeben.