Member-only story

Retrieval-Augmented Generation (RAG) — Basics to Advanced Series (Part 1)

Decoding the bigger picture !!

Chandan Durgia · Follow

4 min read · Jan 24, 2024

-- 1

Photo by Romain Dancre on Unsplash

Pretext: I have been working on RAG build and efficacy assessments for a while and I realised there is certainly a lack of simplified resources available to understand the domain better. That's why, I've am creaing this blog series, where the aim is to share my RAG journey lessons from scratch. If you come across any parts that need further explanation, please leave a comment.

Let's dive in and unravel the mysteries of the RAG value chain! All the best !!

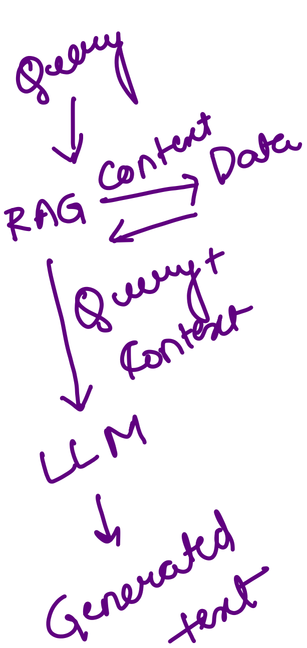
What is RAG and how is this different from ChatGPTs (LLMs)?

ChatGPT (or any other variant) is a Large Language Model (LLM) trained on massive amounts of text data **(from the internet)** and uses deep learning techniques to learn patterns and generate coherent and contextually relevant responses.

However, there are some glaring issues with the LLMs, key ones being — 1 false/out-of-date information on the internet, using non-reliable sources 2 from internet and hallucination 3. To avoid this, it is always prudent to leverage a reliable data source and extract the output from it.

An LLM hallucination occurs when a large language model (LLM) generates a response that is either factually incorrect, nonsensical, or disconnected from the input prompt.

And that's where RAG (Retrieval-Augmented Generation) comes into picture. In order to understand what RAG means, lets fragmenting the keywords Retrieval, Augmented and Generation:



- **Retrieval:** Retrieve relevant information or data from a specific source (pdf, API, database etc.) It involves searching and selecting the most appropriate information to be used in generating a response or output.
- **Augmented:** Augment the query with the retrieved information to enhance and generate more accurate and relevant outputs.
- **Generation:** Produce responses based augmented input or context provided. The language model generates new text that is coherent and relevant to the given input or query.

In other words, Retrieval-Augmented Generation (RAG) is a language model that doesn't rely just on the internet but uses a specific data source instead. It enhances the performance of the model by incorporating a separate knowledge base that is not part of its training data. In other words, RAG allows the language model to provide more relevant and accurate information about a specific domain without needing to be retrained.

Exploring the Inner Workings of RAG: The Architectural view

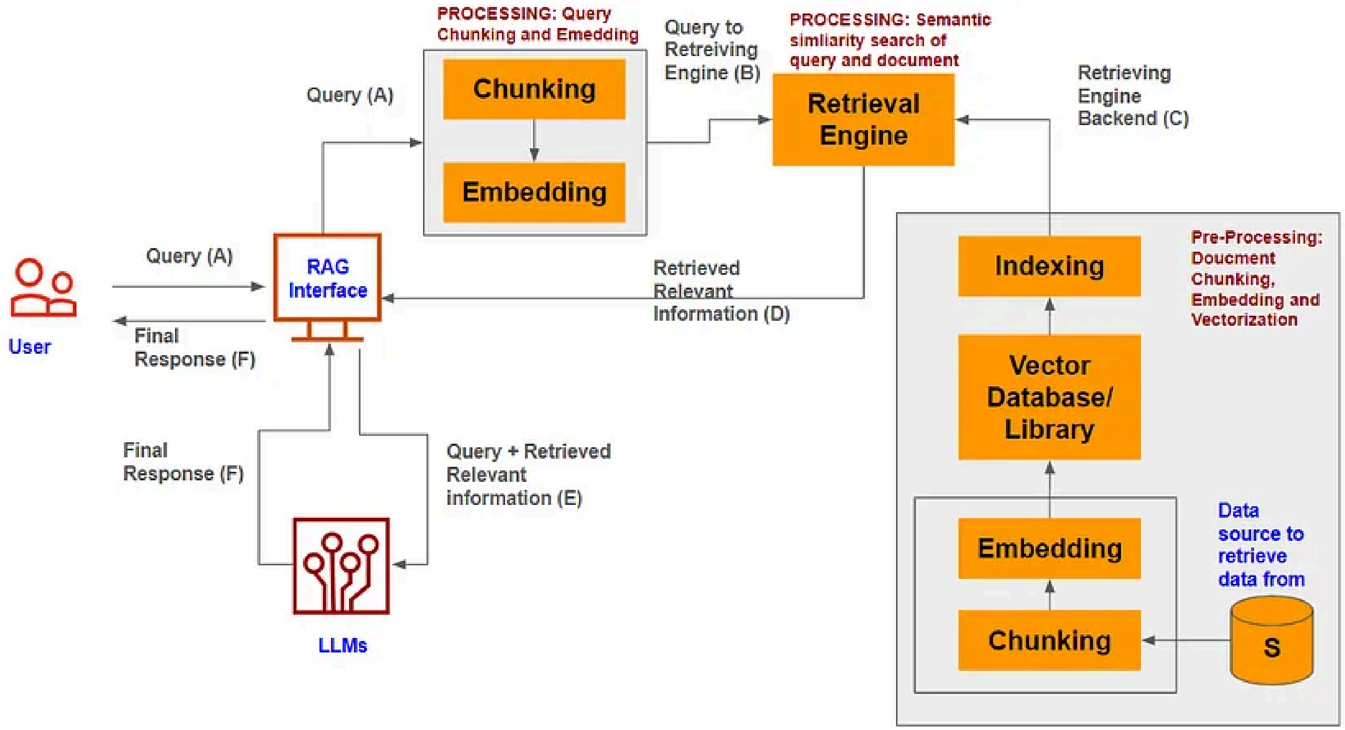

PROCESSING: Query Chunking and Emedding
Chunking
Embedding
Query to Retreiving Engine (B)
PROCESSING: Semantic simliarity search of query and document
Retrieval Engine
Retrieving Engine Backend (C)
Query (A)
Query (A)
User
RAG Interface
Final Response (F)
Retrieved Relevant Information (D)
Pre-Processing: Doucment Chunking, Embedding and Vectorization
Indexing
Vector Database/ Library
Final Response (F)
Query + Retrieved Relevant information (E)
LLMs
Embedding
Chunking
Data source to retrieve data from
S

Image by Author: RAG Architectural View

In this part of the blog, I have provided a comprehensive explanation of the architectural perspective of RAG, providing intricate details and descriptions. (Before you read further, please have a close look at the picture above.)

Key Steps:

1. **Query(A):** The user inputs the query (with a prompt) to the RAG interface. This query could be about either summarising the text or Q&A

from a defined source (S above).

2. **Processing (Chunking and Embedding):** Through the RAG interface the query is sent for processing (chunking and embedding)- which is basically to convert the query text to a mathematical format. (The details on chunking and embedding will be covered later)

3. **Query to Retrieving Engine (B):** The query, now in mathematical form, is sent to the Retrieval Engine.

4. **Processing (Retrieval Engine):** Retrieval engine in simple terms, is a sophisticated context-based search engine which compares the text from the source (S) and the query text. The outcome of the retrieval engine is a text which is similar in context to the query.

5. **Pre-processing:** Note that, expectedly, for the comparison in the Retrieval Engine, the text in the source document (S) also has to be converted into a mathematical form which is done through chunking and embedding. However, in addition, this mathematical form of document (S) is stored in a special vector database and indexed appropriately for faster search. (Don’t get bogged down with the terminologies, the details of each of these components will be explained later)

6. **Retrieved Relevant Information (D)**- The text output from the Retrieval Engine is sent back to the RAG interface.