What is Retrieval-Augmented Generation (RAG)?

Author: Shumin Zhang

What is Retrieval-Augmented Generation?

Retrieval-Augmented Generation (RAG) is the process of enhancing the reference data used by language models (LLMs) through integrating them with traditional information retrieval systems. This hybrid approach allows LLMs to access and utilize external knowledge bases, databases, and other authoritative sources of information, thereby improving the accuracy, relevance, and currency of the generated responses without requiring extensive retraining. Without RAG, LLMs generate responses based on the information they were trained on. With RAG, the response generation process is enriched by integrating external information into the generation.

How does Retrieval-Augmented Generation work?

Retrieval-Augmented Generation works through bringing multiple systems or services to generate the prompt to the LLM. This means there will be required setup to support the different systems and services to feed the appropriate data for a RAG workflow. This involves several key steps:

1. External Data Source Creation:

External data refers to information outside the original training data of the LLM. This data can come from a variety of sources such as APIs, databases, document repositories, and web pages. The data is pre-processed and converted into numerical representations (embeddings) using embedding models, and then stored in a searchable vector database along with reference to the data that was used to generate the embedding. This forms a knowledge library that can be used to augment a prompt when calling into the LLM for generation of a response to a given input.

2. Retrieval of Relevant Information:

When a user inputs a query, it is embedded into a vector representation and matched against the entries in the vector database. The vector database retrieves the most relevant documents or data based on semantic similarity. For example, a query about company leave policies would retrieve both the general leave policy document and the specific role leave policies.

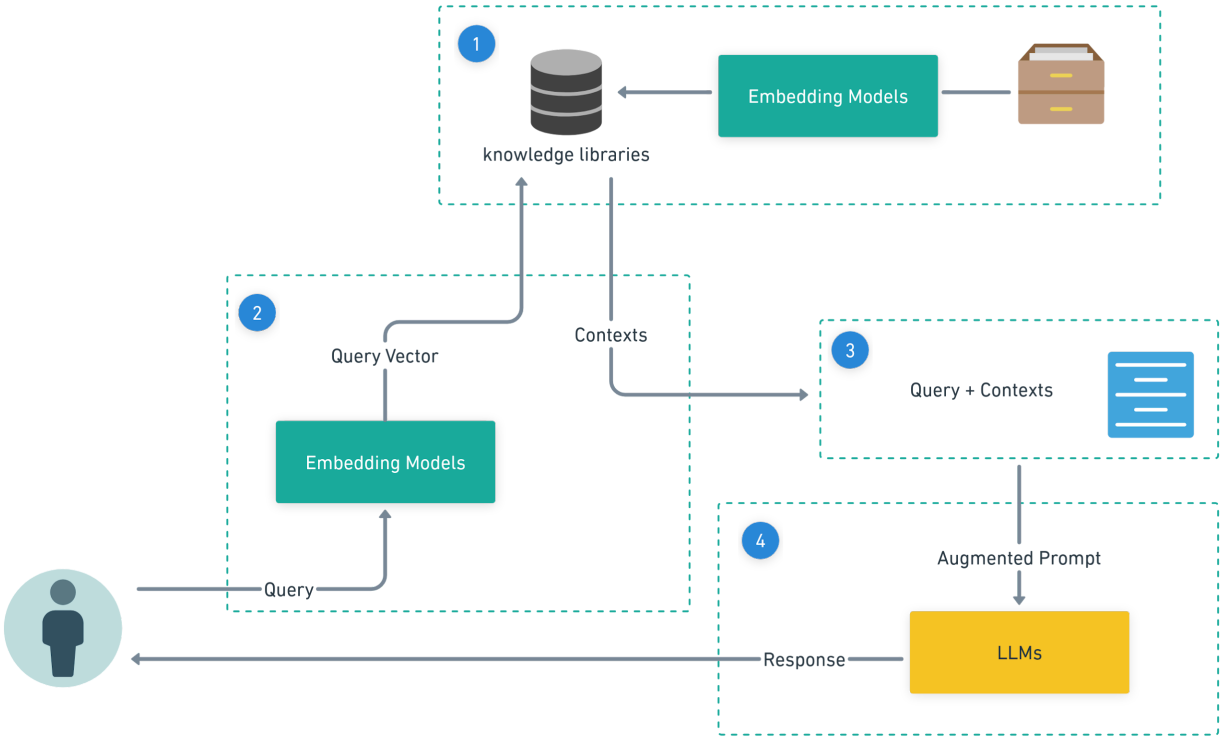
3. Augmentation of LLM Prompt:

The retrieved information is then integrated into the prompt to send to the LLM using prompt engineering techniques. This fully formed prompt is sent to the LLM, providing additional context and relevant data that enables the model to generate more accurate and contextually appropriate responses.

4. Generation of Response:

The LLM processes the augmented prompt and generates a response that is coherent, contextually appropriate, and enriched with accurate, up-to-date information.

The following diagram illustrates the flow of data when using RAG with LLMs.



Why use Retrieval-Augmented Generation?

RAG addresses several inherent challenges of using LLMs by leveraging external data sources:

1. Enhanced Accuracy and Relevance:

By accessing up-to-date and authoritative information, RAG ensures that the generated responses are accurate, specific, and relevant to the user's query. This is particularly important

for applications requiring precise and current information, such as specific company details, release dates and release items, new features available for a product, individual product details, etc..

2. Cost-Effective Implementation:

RAG enables organizations to enhance the performance of LLMs without the need for expensive and time-consuming fine-tuning or custom model training. By incorporating external knowledge libraries, RAG provides a more efficient way to update and expand the model's basis of knowledge.

3. Improved User Trust:

With RAG, responses can include citations or references to the original sources of information, increasing transparency and trust. Users can verify the source of the information, which enhances the credibility and trust of an AI system.

4. Greater Developer Control:

Developers can easily update and manage the external knowledge sources used by the LLM, allowing for flexible adaptation to changing requirements or specific domain needs. This control includes the ability to restrict sensitive information retrieval and ensure the correctness of generated responses. Doing this in conjunction with an evaluation framework (link to evaluation pipeline article) can help to roll out newer content more rapidly to downstream consumers.

Snaplogic GenAI App Builder: Building RAG with Ease

Snaplogic GenAI App Builder empowers business users to create large language model (LLM) powered solutions without requiring any coding skills. This tool provides the fastest path to developing generative enterprise applications by leveraging services from industry leaders such as OpenAI, Azure OpenAI, Amazon Bedrock, Anthropic Claude on AWS, and Google Gemini. Users can effortlessly create LLM applications and workflows using this robust platform.

With Snaplogic GenAI App Builder, you can construct both an indexing pipeline and a Retrieval-Augmented Generation (RAG) pipeline with minimal effort.