RAG

What is Retrieval-Augmented Generation?

Retrieval-Augmented Generation (RAG) is the process of optimizing the output of a large language model, so it references an authoritative knowledge base outside of its training data sources before generating a response. Large Language Models (LLMs) are trained on vast volumes of data and use billions of parameters to generate original output for tasks like answering questions, translating languages, and completing sentences. RAG extends the already powerful capabilities of LLMs to specific domains or an organization's internal knowledge base, all without the need to retrain the model. It is a cost-effective approach to improving LLM output so it remains relevant, accurate, and useful in various contexts.

Why is Retrieval-Augmented Generation important?

LLMs are a key artificial intelligence (AI) technology powering intelligent chatbots and other natural language processing (NLP) applications. The goal is to create bots that can answer user questions in various contexts by cross-referencing authoritative knowledge sources. Unfortunately, the nature of LLM technology introduces unpredictability in LLM responses. Additionally, LLM training data is static and introduces a cut-off date on the knowledge it has.

Known challenges of LLMs include:

- Presenting false information when it does not have the answer.
- Presenting out-of-date or generic information when the user expects a specific, current response.
- Creating a response from non-authoritative sources.
- Creating inaccurate responses due to terminology confusion, wherein different training sources use the same terminology to talk about different things.

You can think of the Large Language Model as an over-enthusiastic new employee who refuses to stay informed with current events but will always answer every question with absolute confidence. Unfortunately, such an attitude can negatively impact user trust and is not something you want your chatbots to emulate!

RAG is one approach to solving some of these challenges. It redirects the LLM to retrieve relevant information from authoritative, pre-determined knowledge sources. Organizations have greater control over the generated text output, and users gain insights into how the LLM generates the response.

What are the benefits of Retrieval-Augmented Generation?

RAG technology brings several benefits to an organization's generative AI efforts.

Cost-effective implementation

Chatbot development typically begins using a foundation model. Foundation models (FMs) are API-accessible LLMs trained on a broad spectrum of generalized and unlabeled data. The computational and financial costs of retraining FMs for organization or domain-specific information are high. RAG is a more cost-effective approach to introducing new data to the LLM. It makes generative artificial intelligence (generative AI) technology more broadly accessible and usable.

Current information

Even if the original training data sources for an LLM are suitable for your needs, it is challenging to maintain relevancy. RAG allows developers to provide the latest research, statistics, or news to the generative models. They can use RAG to connect the LLM directly to live social media feeds, news sites, or other frequently-updated information sources. The LLM can then provide the latest information to the users.

Enhanced user trust

RAG allows the LLM to present accurate information with source attribution. The output can include citations or references to sources. Users can also look up source documents themselves if they require further clarification or more detail. This can increase trust and confidence in your generative AI solution.

More developer control

With RAG, developers can test and improve their chat applications more efficiently. They can control and change the LLM's information sources to adapt to changing requirements or cross-functional usage. Developers can also restrict sensitive information retrieval to different authorization levels and ensure the LLM generates appropriate responses. In addition, they can also troubleshoot and make fixes if the LLM references incorrect information sources for specific questions. Organizations can implement generative AI technology more confidently for a broader range of applications.

How does Retrieval-Augmented Generation work?

Without RAG, the LLM takes the user input and creates a response based on information it was trained on—or what it already knows. With RAG, an information retrieval component is introduced that utilizes the user input to first pull information from a new data source. The user query and the relevant information are both given to the LLM. The LLM uses the new knowledge and its training data to create better responses. The following sections provide an overview of the process.

Create external data

The new data outside of the LLM's original training data set is called *external data*. It can come from multiple data sources, such as a APIs, databases, or document repositories. The data may exist in various formats like files, database records, or long-form text. Another AI technique, called *embedding language models*, converts data into numerical representations and stores it in a vector database. This process creates a knowledge library that the generative AI models can understand.

Retrieve relevant information

The next step is to perform a relevancy search. The user query is converted to a vector representation and matched with the vector databases. For example, consider a smart chatbot that can answer human resource questions for an organization. If an employee searches, *"How much annual leave do I have?"* the system will retrieve annual leave policy documents alongside the individual employee's past leave record. These specific documents will be returned because they are highly-relevant to what the employee has input. The relevancy was calculated and established using mathematical vector calculations and representations.

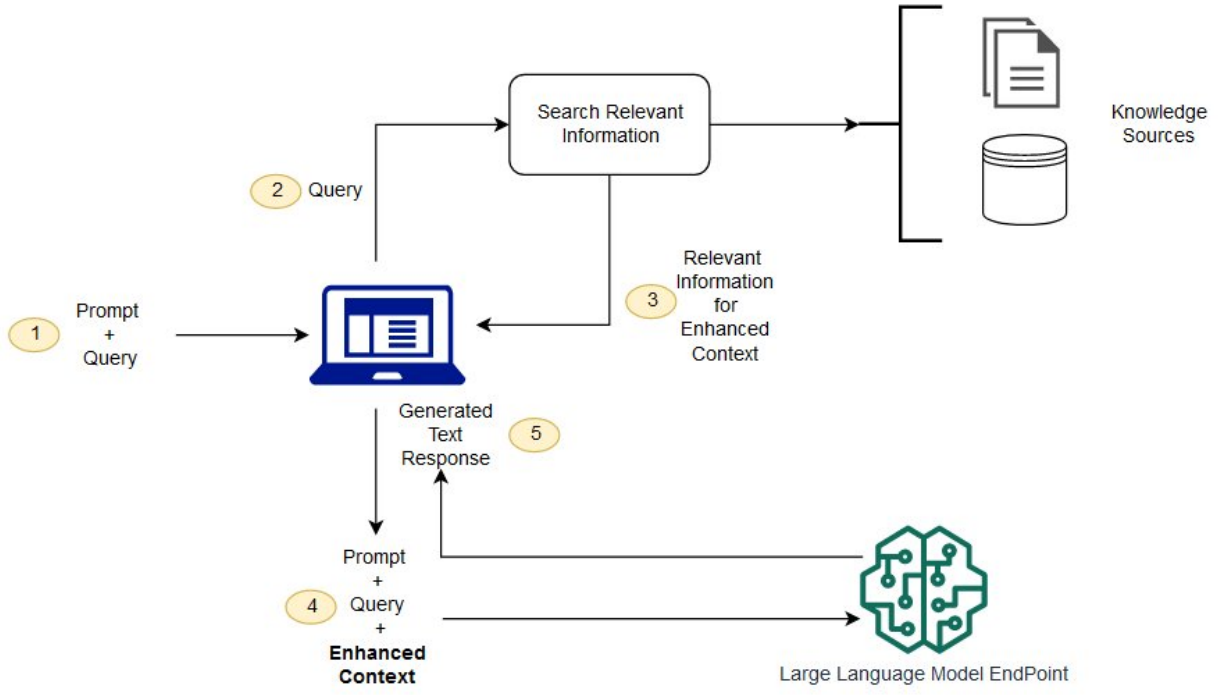
Augment the LLM prompt

Next, the RAG model augments the user input (or prompts) by adding the relevant retrieved data in context. This step uses prompt engineering techniques to communicate effectively with the LLM. The augmented prompt allows the large language models to generate an accurate answer to user queries.

Update external data

The next question may be—what if the external data becomes stale? To maintain current information for retrieval, asynchronously update the documents and update embedding representation of the documents. You can do this through automated real-time processes or periodic batch processing. This is a common challenge in data analytics—different data-science approaches to change management can be used.

The following diagram shows the conceptual flow of using RAG with LLMs.


Search Relevant Information
Knowledge Sources
2 Query
Relevant Information for Enhanced Context
3
1
Prompt + Query
Generated Text Response
5
Prompt + Query + Enhanced Context
4
Large Language Model EndPoint

What is the difference between Retrieval-Augmented Generation and semantic search?

Semantic search enhances RAG results for organizations wanting to add vast external knowledge sources to their LLM applications. Modern enterprises store vast amounts of information like manuals, FAQs, research reports, customer service guides, and human resource document repositories across various systems. Context retrieval is challenging at scale and consequently lowers generative output quality.

Semantic search technologies can scan large databases of disparate information and retrieve data more accurately. For example, they can answer questions such as, *"How much was spent on machinery repairs last year?"* by mapping the question to the relevant documents and returning specific text instead of search results. Developers can then use that answer to provide more context to the LLM.

Conventional or keyword search solutions in RAG produce limited results for knowledge-intensive tasks. Developers must also deal with word embeddings, document chunking, and other complexities as they manually prepare their data. In contrast, semantic search technologies do all the work of knowledge base preparation so developers don't have to. They also generate semantically relevant passages and token words ordered by relevance to maximize the quality of the RAG payload.

How can AWS support your Retrieval-Augmented Generation requirements?

Amazon Bedrock is a fully-managed service that offers a choice of high-performing foundation models—along with a broad set of capabilities—to build generative AI applications while simplifying development and maintaining privacy and security. With knowledge bases for Amazon Bedrock, you can connect FMs to your data sources for RAG in just a few clicks. Vector conversions, retrievals, and improved output generation are all handled automatically.

For organizations managing their own RAG, Amazon Kendra is a highly-accurate enterprise search service powered by machine learning. It provides an optimized Kendra Retrieve API that you can use with Amazon Kendra’s high-accuracy semantic ranker as an enterprise retriever for your RAG workflows. For example, with the Retrieve API, you can:

- Retrieve up to 100 semantically-relevant passages of up to 200 token words each, ordered by relevance.
- Use pre-built connectors to popular data technologies like Amazon Simple Storage Service, SharePoint, Confluence, and other websites.
- Support a wide range of document formats such as HTML, Word, PowerPoint, PDF, Excel, and text files.
- Filter responses based on those documents that the end-user permissions allow.

Amazon also offers options for organizations who want to build more custom generative AI solutions. Amazon SageMaker JumpStart is a ML hub with FMs, built-in algorithms, and prebuilt ML solutions that you can deploy with just a few clicks. You can speed up RAG implementation by referring to existing SageMaker notebooks and code examples.

Get started with Retrieval-Augmented Generation on AWS by creating a free account today