**Retrieval-Augmented Generation**

**What is Retrieval-Augmented Generation?**

Retrieval-Augmented Generation (RAG) enhances the output of a large language model by incorporating information from an external, authoritative knowledge base beyond its original training data. Large Language Models (LLMs) are trained on massive datasets and utilize billions of parameters to perform tasks such as answering questions, translating languages, and completing sentences. RAG builds on the strengths of LLMs by enabling them to access specific domain knowledge or an organization's internal data without requiring retraining. This approach is a cost-effective way to improve the relevance, accuracy, and usefulness of LLM-generated responses across various applications.

**How Does Retrieval-Augmented Generation (RAG) Work?**

Without RAG, a Large Language Model (LLM) generates responses based solely on the information it was trained on—essentially, what it already knows. In contrast, RAG introduces an information retrieval component that first searches external data sources using the user’s input. The retrieved information, along with the user query, is then provided to the LLM, enabling it to generate more accurate and contextually relevant responses. The following steps outline this process:

**1. Creating External Data**

External data refers to any information beyond the LLM’s original training dataset. It can be sourced from APIs, databases, or document repositories and may exist in various formats, such as files, structured records, or long-form text. To make this data usable for AI models, a technique called **embedding language models** converts it into numerical representations, which are stored in a **vector database**—effectively creating a structured knowledge library for retrieval.

**2. Retrieving Relevant Information**

When a user submits a query, the system performs a **relevancy search** by converting the query into a vector representation and matching it against stored vectors in the database. For example, in an HR chatbot scenario, if an employee asks, *"How much annual leave do I have?"*, the system retrieves relevant documents, such as the company’s leave policy and the employee’s personal leave records. This retrieval is based on mathematical vector calculations that determine the most relevant information.

**3. Augmenting the LLM Prompt**

Once the relevant information is retrieved, it is added to the user query before being processed by the LLM. This step, known as **prompt augmentation**, ensures that the LLM generates responses based on both its trained knowledge and the newly retrieved data. **Prompt engineering** techniques help structure the input effectively to maximize response accuracy.

**4. Updating External Data**

To ensure the retrieved information remains current, external data must be updated regularly. This can be done through **automated real-time updates** or **periodic batch processing** to refresh both documents and their corresponding vector embeddings. Managing evolving data is a common challenge in data science, and various change management techniques can be applied to keep the knowledge base accurate and up to date.

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

A diagram of a computer process

AI-generated content may be incorrect.

Ref: <https://aws.amazon.com/what-is/retrieval-augmented-generation/#:~:text=Retrieval%2DAugmented%20Generation%20(RAG),sources%20before%20generating%20a%20response>.