

RAG vs Agentic RAG vs CAG

A comprehensive review about these three methods for enhancing LLM outputs.



Andrés Herencia | Data Scientist & Machine Learning Engineer



ABNASIA.ORG

Introduction

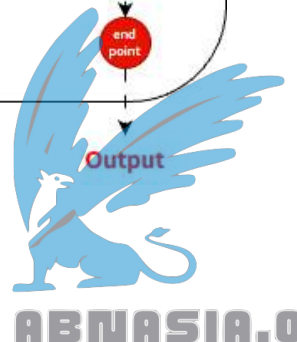
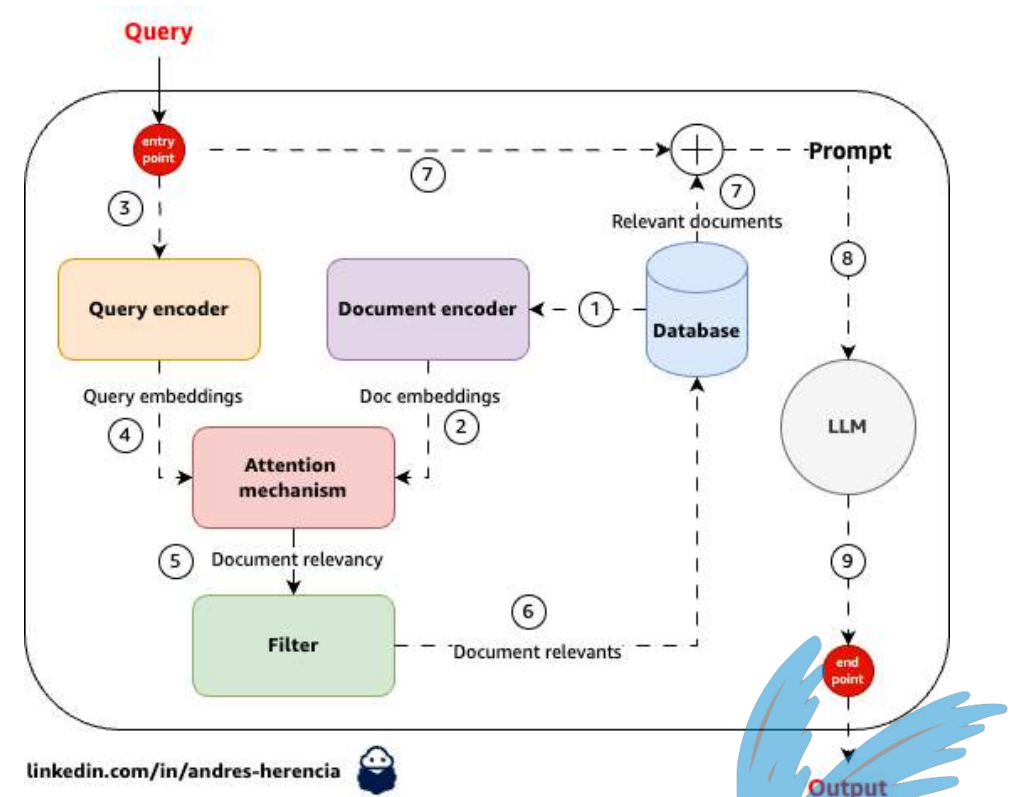
- The emergence of **Generative AI** and **Large Language Models (LLMs)** has revolutionized **Natural Language Processing (NLP)**.
- These models enable **applications** such as:
 - **Interactive conversational** tools.
 - Intelligent **search engines**.
 - Automated **response generation**.
- However, LLMs are limited by the static knowledge encoded during their training.
- To overcome these limitations, **Retrieval-Augmented Generation (RAG)** systems were introduced.



RAG systems

2

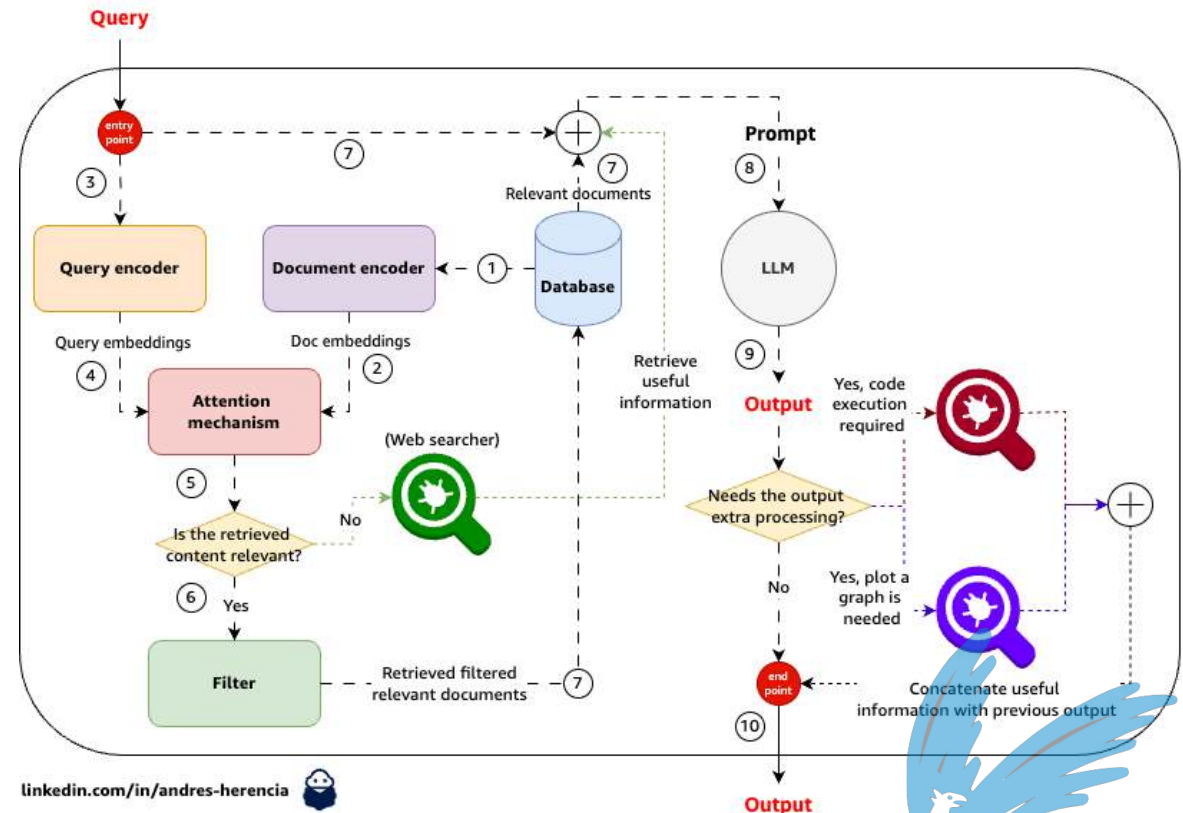
- RAG stands for **Retrieval Augmented Generation**.
- **Core Mechanism:** Combines a retrieval system with a generative model.
 - **Retrieves** missing information relevant to the user query.
 - Integrates (**augment**) the retrieved data into the input for the generative model.
 - Enables **generation** of more comprehensive and context-aware responses.
- **Memory Types:**
 - **Parametric Memory:** Knowledge encoded during the training of the generative model.
 - **External Memory:** Non-parametric memory, dynamically retrieved from databases or sources.



Agentic RAG systems

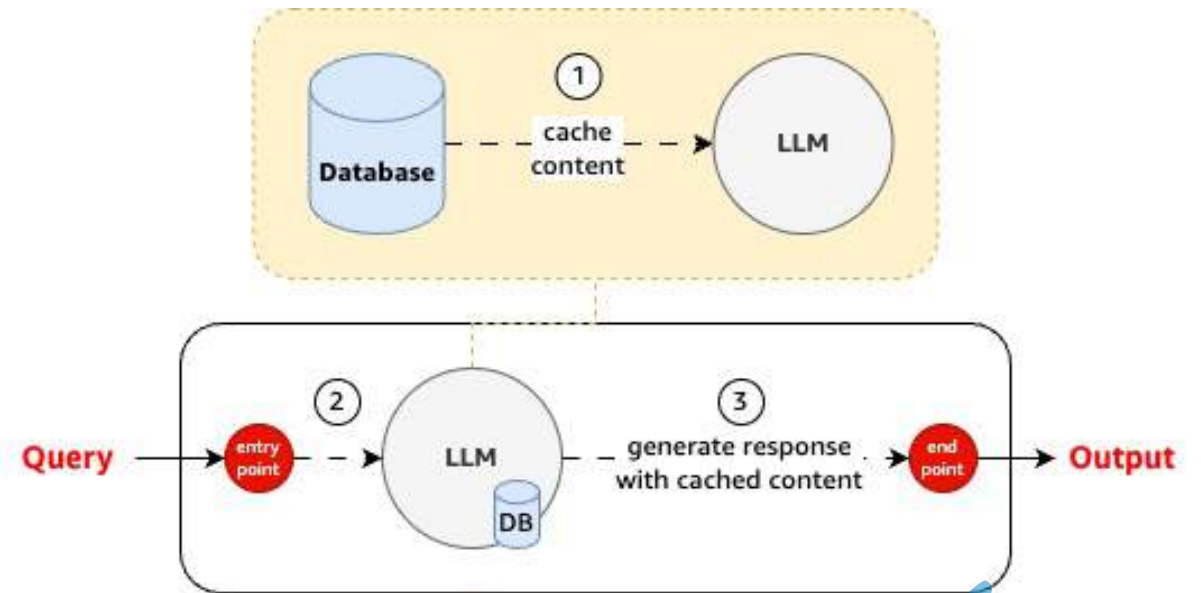
3

- **Core Framework:** Builds upon RAG systems by integrating agents as modular components to enhance functionality.
- **Capabilities of Agents:**
 - Interact with external systems (e.g., perform real-time web searches or access APIs).
 - Retrieve dynamic data such as weather updates or financial metrics.
 - Execute complex tasks like running code or simulations.
- **Decision-Making Module:**
 - Evaluates user queries.
 - Activates specific agents as needed to handle the task.



Cache-Augmented Generation Systems

- **Purpose:** Minimizes latency while maintaining response precision by replacing retrieval with a caching mechanism.
- **How It Works:**
 - **KV Cache:** Stores document information in the LLM's memory (context window).
 - **Inference:** Uses precomputed attention values, processing only the user's query.
 - **Reset Mechanism:** Truncates token additions to maintain efficiency.
- This architecture **eliminates retrieval latency** and streamlines response generation.



[linkedin.com/in/andres-herencia](https://www.linkedin.com/in/andres-herencia)



ABNASIA.ORG

RAG vs Agentic RAG vs CAG

5

System	Advantages	Disadvantages
RAG	Simple design	Limited in handling complex tasks
	Effective for general-purpose retrieval	Cannot integrate external processes
	Well-suited for tasks with manageable latency	Suffers from retrieval latency and accuracy dependency
Agentic RAG	Extends RAG with agents for advanced tasks	Increased architectural complexity
	Improves adaptability for complex workflows	Higher latency due to agent interaction
		Greater dependency on coordination between agents
CAG	Removes retrieval steps	Constrained by memory limitations
	Minimizes latency	Less flexible for dynamic or updated knowledge
	Simplifies execution by relying on preloaded and cached data	Struggles with scalability for extensive datasets



Use cases

6

- **RAG:**

- When you need to use the LLM with information about external sources
- When you need up-to-date information to answer user's queries.

- **Agentic RAG:**

- When you need to complement your output with some operations (code execution, web searches, post-processing, etc.)
- When you need to connect your RAG with other 3rd party tools in an autonomous way.

- **CAG:**

- When latency times are critical in your LLM-based application.
- When you want to use RAG and your external database is relatively small.

