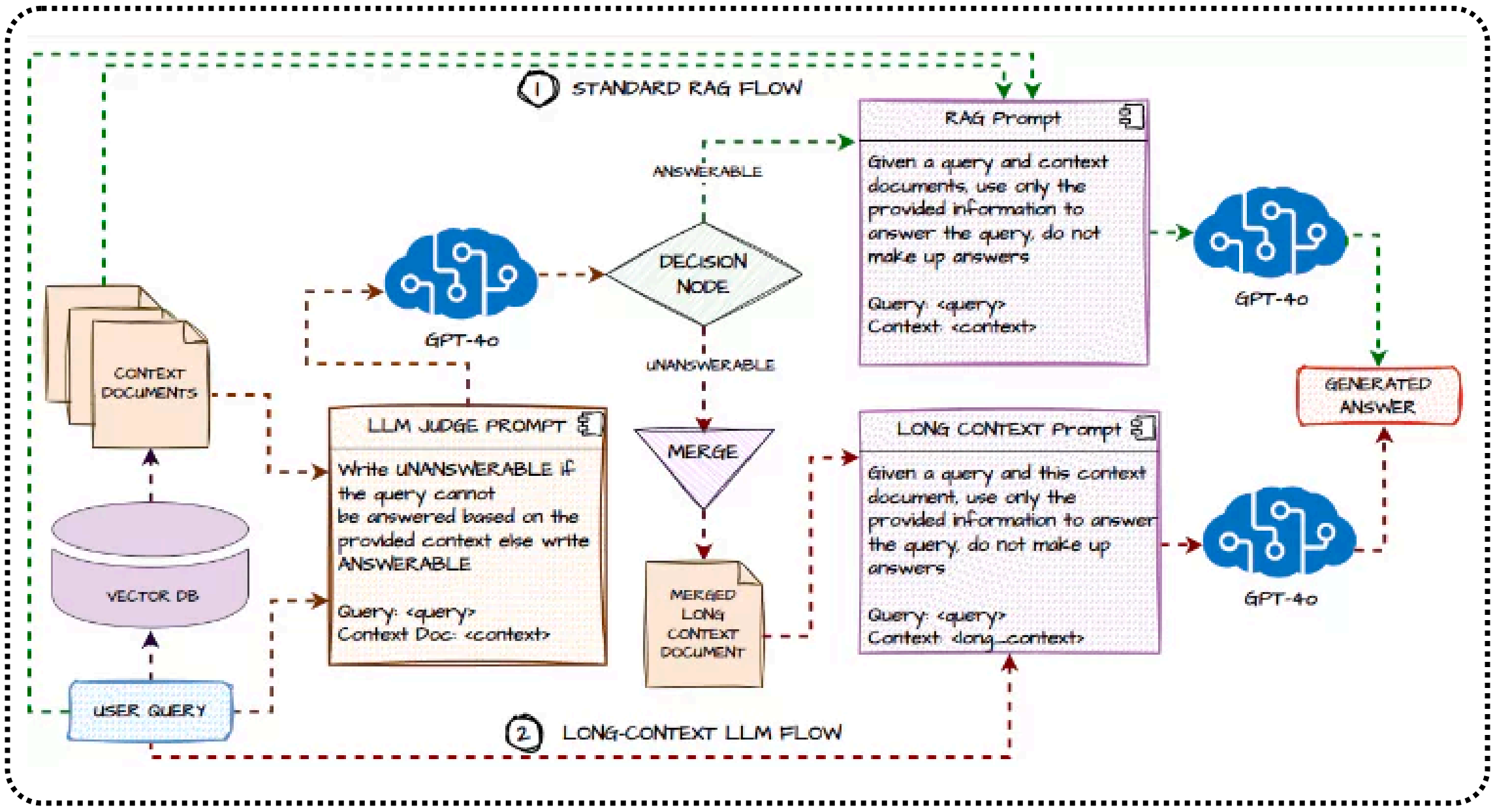
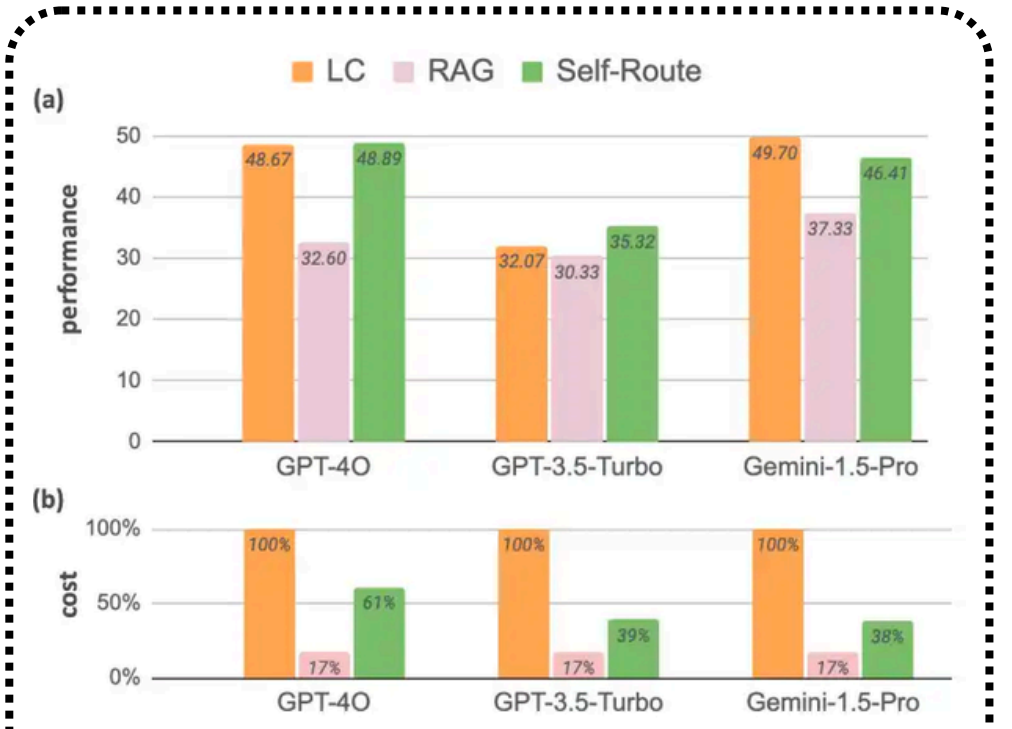
Evolution of RAG, Long Context LLMs to Agentic RAG



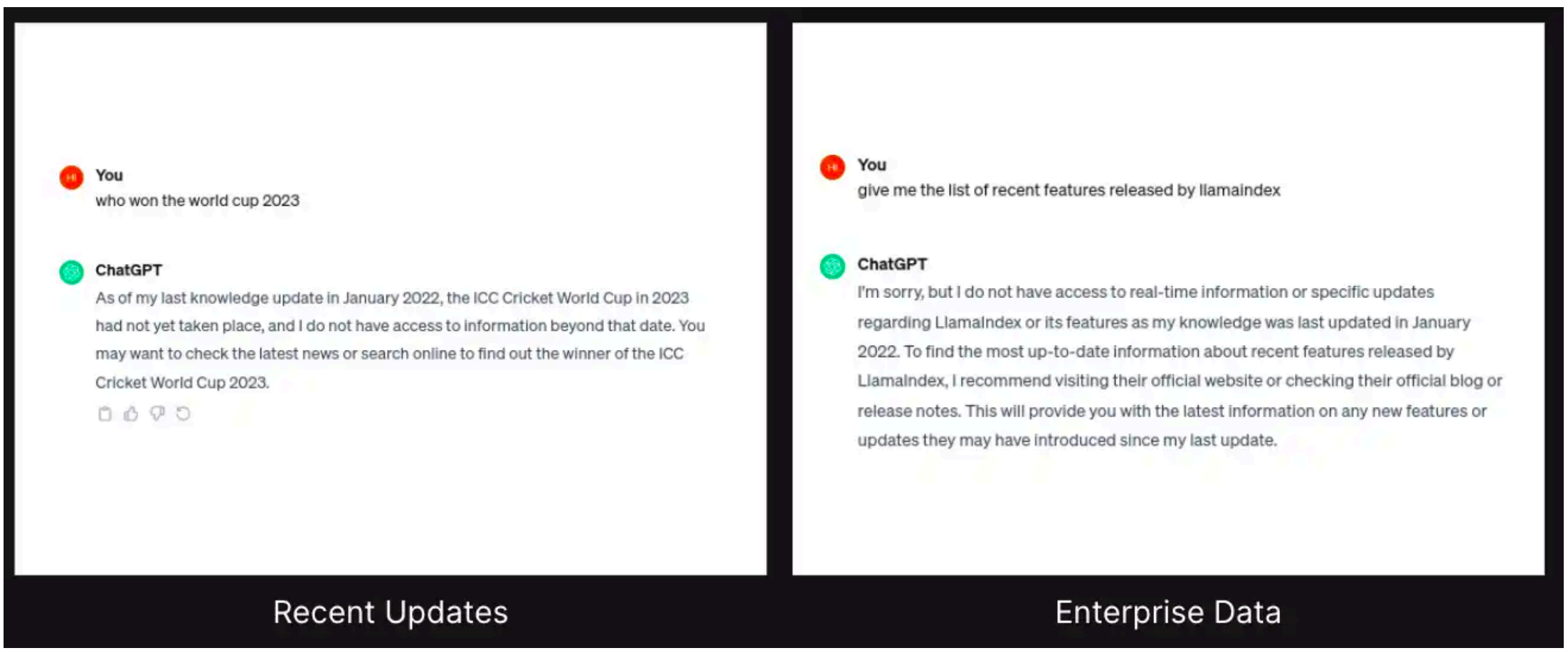
Feature	Long Context LLMs	RAG (Retrieval Augmented Generation)	Agentic RAG
Core Components	Static knowledge base	LLM+ External data source	LLM+ Retrieval module + Autonomous Agent
Information Retrieval	No external retrieval	Queries external data sources during responses	Queries external databases and select appropriate tool
Interaction Capability	Limited to text generation	Retrieves and integrates context	Autonomous decisions to take actions
Use Cases	Text summarization, understanding	Augmented responses and contextual generation	Multi-tasking, end-to-end task generation



Evolution of Agentic RAG, So Far

When large language models (LLMs) emerged, they revolutionized how people engaged with information.

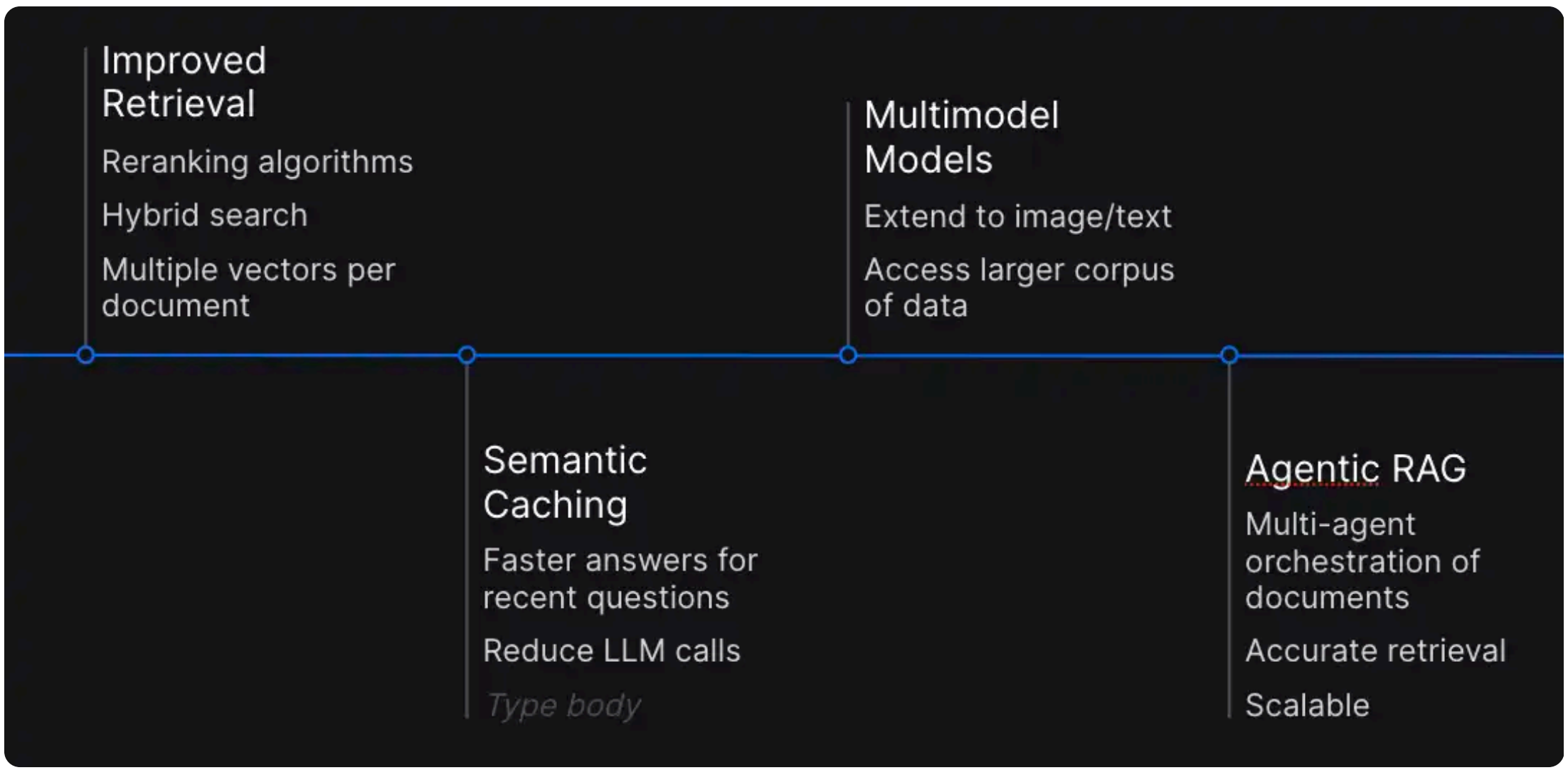
However, it was noted that relying on them to solve complex problems sometimes led to factual inaccuracies, as they depend entirely on their internal knowledge base. This led to the rise of the Retrieval-Augmented Generation (RAG).



RAG is a technique or a methodology to augment the external knowledge into the LLMs. We can directly connect the external knowledge base to LLMs, like chat GPT, and prompt the LLMs to fetch answers about the external knowledge base.

Agentic RAG and Agentic AI are closely related terms that fall under the broader umbrella of Agentic Systems.

Before we study Agentic RAG in detail, let's look at the recent discoveries in the fields of LLM and RAG.



- **Improved Retrieval**: It is important to optimize retrieval for continuous performance. Recent developments focus on reranking algorithms and hybrid search methodologies, also employing multiple vectors per document to enhance relevance identification.

- **Semantic Caching**: Semantic caching has emerged as a key strategy to mitigate computational complexity. It allows storing answers to the recent queries which can be used to answer the similar requests without repeating.

- **Multimodal Integration**: This expands the capabilities of LLMs and RAG beyond text, integrating images and other modalities. This integration facilitates seamless integration between textual and visual data.

Key Differences between RAG and AI Agents

Features	RAG	AI Agent
Primary Focus	Knowledge Augmentation	Action and Interaction
Mechanism	Information Retrieval and Integration	Tool Utilization and Decision-making
Strengths	Improved Accuracy, Reliability and Domain Expertise	Task completion, Problem Solving
Limitations	Retrieved Performance, Static Context	Tool Dependency, Complexity of Agentic Design

These comparisons help us understand how these advanced technologies differ in their approach to augmenting and performing tasks.

- Primary Focus: The primary goal of RAG systems is to augment knowledge, which consists of a model's understanding by retrieving relevant information. This allows for more decision-making and improved contextual understanding. In contrast, AI agents are designed for actions and environmental interactions. Here, agents go a step ahead and interact with the tools and complete complex tasks.

- **Mechanisms**: RAG depends on information extraction and integration. It pulls data from external sources and integrates it into the responses, whereas AI agents function through tool utilization and autonomous decision-making.

- **Strength**: RAG's strength lies in its ability to provide improved responses. By connecting LLM with external data, RAG prompts to provide more accurate and contextual information. Agents, on the other hand, are masters at task execution autonomously by interacting with the environment.

- **Limitations**: RAG systems face challenges like retrieval problems, static context, and a lack of autonomous intervention while generating responses. Despite countless strengths, agents' major limitations include solely depending on tools and the complexity of agentic design patterns.

Architectural Difference Between Long Context LLMs, RAGs and Agentic RAG

So far, you have observed how integrating LLMs with the retrieval mechanisms has led to more advanced AI applications and how Agentic RAG (ARAG) is optimizing the interaction between the retrieval system and the generation model.

Now, backed by these learnings, let's explore the architectural differences to understand how these technologies build upon each other.

Feature	Long Context LLMs	RAG (Retrieval Augmented Generation)	Agentic RAG
Core Components	Static knowledge base	LLM+ External data source	LLM+ Retrieval module + Autonomous Agent
Information Retrieval	No external retrieval	Queries external data sources during responses	Queries external databases and select appropriate tool
Interaction Capability	Limited to text generation	Retrieves and integrates context	Autonomous decisions to take actions
Use Cases	Text summarization, understanding	Augmented responses and contextual generation	Multi-tasking, end-to-end task generation

Architectural Differences

Long Context LLMs: Transformer-based models such as GPT -3 are usually trained on a large amount of data and rely on a static knowledge base. Their architecture is compatible for text generation and summarization, where they do not require external information to generate responses. However, they lack the susceptibility to provide updated or specialized knowledge. Our area of focus is the Long Context LLM models. These models are designed to handle and process much longer input tokens compared to traditional LLMs.

Models such as GPT-3 or earlier models are often limited to the number of input tokens.

Long context models address such limitations by extending the context window size, making them better at:

- Summarizing larger documents
- Maintaining coherence over long dialogues
- Processing documents with extensive context

RAG (Retrieval Augmented Generation): RAG has emerged as a solution to overcome LLMs' limitations. The retrieval component allows LLMs to be connected to external data sources, and the augmentation component allows RAG to provide more contextual information than a standard LLM. However, RAG still lacks autonomous decision-making capabilities.

Agentic RAG: Next is Agentic RAG, which incorporates an additional intelligence layer. It can retrieve external information and includes an autonomous reasoning module that analyzes the retrieved information and implements strategic decisions.

These architectural distinctions help explain how each system allows knowledge, augmentation, and decision-making differently. Now comes the point where we need to determine the most suitable—LLMs, RAG, and Agentic RAG. To pick one, you need to consider specific requirements such as Cost, Performance, and Functionality.

For more information, kindly visit this article

Advanced AI Agents LLMs

Evolution of RAG, Long Context LLMs to Agentic RAG

Evolution of Agentic RAG: Long Context LLMs, RAG, and Agentic RAG for dynamic task execution, balancing cost with Self-Route fusion.

Sushant Thakur 18 Mar, 2025

Long context models address such limitations by extending the context window size, making them better at:

- Summarizing larger documents
- Maintaining coherence over long dialogues
- Processing documents with extensive context

RAG (Retrieval Augmented Generation): RAG has emerged as a solution to overcome LLMs' limitations. The retrieval component allows LLMs to be connected to external data sources, and the augmentation component allows RAG to provide more contextual information than a standard LLM. However, RAG still lacks autonomous decision-making capabilities.

Agentic RAG: Next is Agentic RAG, which incorporates an additional intelligence layer. It can retrieve external information and includes an autonomous reasoning module that analyzes the retrieved information and implements strategic decisions.

These architectural distinctions help explain how each system allows knowledge, augmentation, and decision-making differently. Now comes the point where we need to determine the most suitable—LLMs, RAG, and Agentic RAG. To pick one, you need to consider specific requirements such as Cost, Performance, and Functionality.