ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Neural Graph Retrieval for Intelligent Enterprise Agents

Mallesham Goli
Independent Researcher
mallesham.goli.research01@gmail.com

Abstract

Graph-Enhanced Retrieval-Augmented Generation (Graph-RAG) and contextual architectures for large language models (LLMs) deliver performance, reliability, and governance for enterprise applications. Graph-enhanced RAG addresses neural retrieval deficiencies within enterprise chatbots, question-answering systems, and service automation agents. A contextually aware agent-centric architecture combines knowledge graphs, domain ontologies, and personal models to ensure contextual accuracy and suitability for the use case. In contrast to standard RAG augmentation, application-specific data and reasoning mechanisms are connected through prompt engineering, plugins, and external tools. Evaluation methodologies for context-aware agents include dedicated benchmark suites and task-centric quality metrics.

Enterprise deployments of RAG and large language models require performance, reliability, and governance. LLM generative capability is appealing for data indexing tasks like customer support chat interactions, service delivery, and knowledge management. However, the neural retrieval component of RAG suffers from novelty, specificity, and appropriateness issues that impact overall user experience. Scaling chatbots to handle repetitive queries created by less elaborate agents leads to a reduction in customer engagement. Operator confidence decreases further when these systems fail to escalate correctly when faced with complex problems.

Keywords: Graph-Enhanced RAG, Context-Aware LLMs, Enterprise AI Systems, Knowledge Graph Integration, Domain Ontologies, Agent-Centric Architectures, Neural Retrieval Optimization, Contextual Reasoning, Prompt Engineering, Tool-Augmented LLMs, Intelligent Service Agents, LLM Governance, AI Reliability, Retrieval Quality, Enterprise Chatbots, Question Answering Systems, Task-Centric Evaluation, AI Benchmarking, Customer Support Automation, AI Performance Optimization.

1. Introduction

In the field of Graph-Enhanced Retrieval-Augmented Generation (Graph-RAG) and contextual Large Language Model (LLM) architectures, the foundations of Retrieval-Augmented Generation (RAG) in enterprise contexts consist of knowledge sources, their treatment in dedicated retrieval pipelines, and latency considerations. RAG is a technique for combining

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

generative LLMs with search systems, augmenting generation with retrieval results from semantically similar yet largely disjoint data spaces. Result quality is critical in determining output. Nonetheless, for many applications, mere retrievability is less important than retrievability by a suitable query (non-specialists typically state input as free text). Enabling users to derive what they wish from such systems—beyond simple fact retrieval—calls for capability-rich contextual knowledge.

Agents, which serve user requests, have thus far relied on input history, the training set, and propagation of context during conversational exchanges. Enabling intelligent agents also to utilize contextual knowledge akin to a knowledge graph via the addition of a KMS. Within such a KMS, the knowledge structure is derived from an enterprise-wide data structure such as the Enterprise Ontology or dynamically generated during processing and retained until it becomes stale or undesirable. These ontologies are then used to constrain any dialog with the Web LLM (or a specific tool). Special (non-consumer) LLMs with specific scope and context may also receive guidance input to ensure suitability and safety of output. Graph-enhanced contextual LLM-architecture support for Knowledge and Service Delivery Automation agents is also considered.

2. Foundations of Graph-Enhanced RAG

Contextual LLM architectures incorporate knowledge retrieval to manage the scaling problems of pure LLM architectures. Knowledge Retrieval Augmented Generation (RAG) refers to a specific integration pattern for LLM and knowledge management components. RAG is particularly suitable for enterprise scenarios where the open-ended nature of LLMs is better controlled using an additional layer of information retrieval. A dedicated retrieval mechanism permits the use of dedicated data sources (e.g., an unearthed information space or highly optimized KBs), provides a faster response latency, supports versioned configuration, and is aligned with specific governance policies. More generally, RAG enables the retrieval of narrower information segments, thus reducing the input prompt required for a specific application task.

Information-Retrieval Augmented Generation (RAG) is a special case of the broader Retrieval-Augmented Generation approach, which allows context-serving LLMs to obtain information from external sources. RAG typically wraps GPT-like LLMs and is tailored for specific tasks through a dedicated retrieval mechanism. Improvements in prompting, generation quality, latency, and governance are attributable to the retrieval-enhanced architecture. In enterprise environments, however, the improvement scale may be diminished due to closed

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

information spaces and separate RAG training. The Google Turing NLG model also introduces a RAG-like approach for dialogue management with its LaMDA model. Companies such as Cohere.ai are providing tailor-made LLMs for enterprises using proprietary or prior-open datasets.

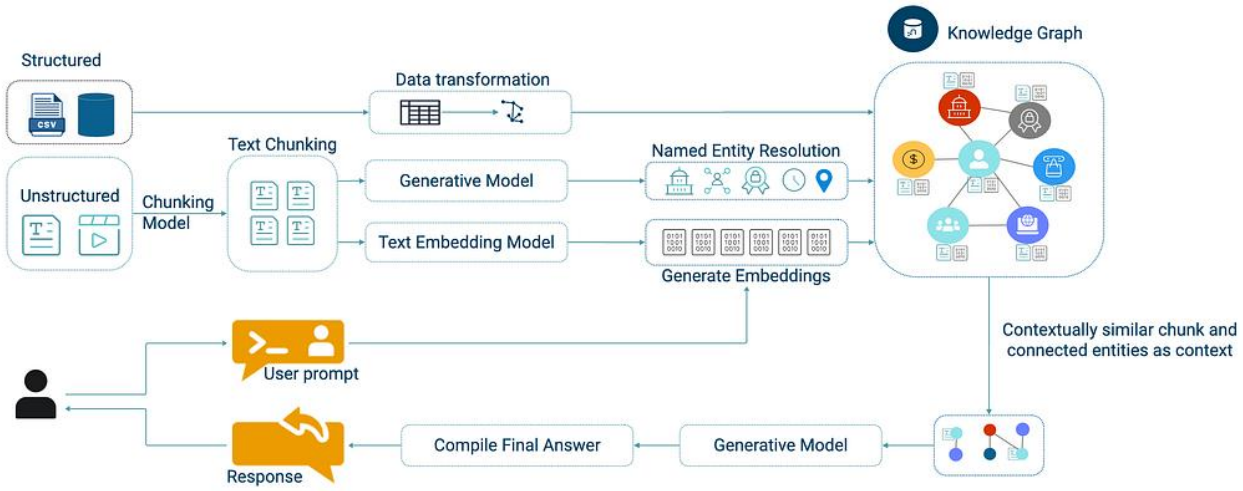


Fig 1: GraphRAG and Agentic Architecture

2.1. Retrieval-Augmented Generation in Enterprise Contexts

In enterprise scenarios, Retrieval-Augmented Generation (RAG)—and Graph-Enhanced RAG (Graph-RAG) in particular—leverage retrieval pipelines that continuously interrogate data sources independent of the generative engine. However, archiving the majority of documents as persistent static JSON or semantic embeddings may degrade or extend retrieval latencies. Several strategies minimize the risk of reduced-quality retrieval augmentations while keeping response latencies within reasonable bounds. RAG pipelines that augment generative models with retrieval augmentations from more easily understandable data sources such as knowledge graphs, backend databases, and curator-approved documents are anticipated to work especially well.

A precise taxonomy-based user query, implicit or explicit, enables a request to be matched against an active knowledge graph, ontology, or taxonomy. Graph traversals such as those enabled by a knowledge database can also deliver contextually relevant subgraphs with

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

comparatively lower overhead than the machine learning models used for semantic search. Contextual knowledge management modules reactively shape context models based on user behaviors to optimize the quality of augmentations offered by relevant sources.

2.2. Graph-Structured Knowledge and Reasoning

Graph-Structured Knowledge and Reasoning considers enhanced retrieval quality and the richness of contextual knowledge injected into the response generation process. Consider the auxiliary knowledge needed to assist in RAG completion tasks, possibly in interaction with the knowledge modeling and management component introduced previously. Knowledge profiling, organization, and management already leverage knowledge graphs (KGS) to some extent. Transforming the static context-trimming task from RAG into a more elaborate context-guiding task combining static and dynamic elements. The static aspect is rendered by knowledge profiling and the support related to the use of a domain ontology or an enterprise taxonomy of types. The dynamic aspect is given by the full Knowledge-Retrieval-Assisted Generation (KRAG) candidate making the description of the methods richer and clearer.

Context-augmented reasoning relies on the definition of the User, Role, and Domain Ontology associated with a user request. The interaction with an enterprise knowledge graph and ontology-based treachery for enabling user- and role-images matching with ontologies to be combined with entity-enriched CK are then considered. The KGS may introduce further sources of information for augmenting LLM responses generated by the regular context-based. Given the provenance-traced nature of knowledge graphs and ontologies and the possibility of associating with the multiplication of the static context by substituting the User, Role, and Domain Ontology of the profile-traced CK, help in guiding the generation process by ranking candidate responses relative to the content of the Knowledge graph to promote novelty and relevance w.r.t. the CK.

Equation 1: Retrieval candidate score in Graph-RAG

- relevance,
- novelty,
- diversity,
- contextual fit,

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

- and latency/governance trade-offs.

So define the score of a candidate x_i as a weighted sum.

Step 1: start from the contributing factors

$$S_i \propto \text{Relevance} + \text{Novelty} + \text{Diversity} + \text{Context Fit} - \text{Penalty}$$

Step 2: convert proportional form into weighted linear form

$$S_i = \alpha R_i + \beta N_i + \gamma D_i + \delta C_i - \lambda L_i$$

where:

- R_i = semantic relevance of candidate i to query q
- N_i = novelty of candidate i
- D_i = diversity contribution
- C_i = context fitness to user/role/domain
- L_i = latency or computational penalty
- $\alpha, \beta, \gamma, \delta, \lambda \geq 0$

Step 3: normalize weights if desired

For interpretability, enforce:

$$\alpha + \beta + \gamma + \delta + \lambda = 1$$

Then S_i becomes a normalized enterprise retrieval score.

Step 4: ranking rule

Candidates are ranked by:

$$x_{(1)}, x_{(2)}, \ldots = \underset{i}{\text{sort}}(S_i, \text{descending})$$

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

3. Methodology

The presented architecture complements enterprise AI agents' system architecture by emphasizing functional decomposition principles for the LLM component and various associated modules. Modularity promotes internal development and publishing in open-source standards, enabling third-party capabilities via established interfaces and contributing towards agent-centric AI paradigms. Five principles underpin this approach: 1) Modular elements encapsulate specific behavior, enabling independent adaptation and impact testing; 2) Compatibility with heterogeneous enterprise AI systems fosters ubiquitous deployment; 3) Security-by-design principles bolster trustworthy AI; 4) Pre-release benchmarking ensures readiness for target contexts; 5) Published artifacts form part of a dedicated research benchmark suite.

The specialized context-aware modules surface within the modular LLM integration framework. Three representative plugin archetypes are defined: Prompt Engineering applies structured prompt tuning in conjunction with LLM context windows, Steering applies context augmentation to guide model behavior in a relevant, consistent, and safe manner, and Tooling applies a typed interface for various interfacing models for code, image, other media generation, and API-querying tasks, supporting parallel execution. These patterns address distinct but overlapping considerations in device and LLM usage whilst remaining compatible with agent-centric development methodologies, supporting sequential, hierarchical, and parallel LLM agent integration patterns.

Motivations for such a functional decomposition arise from the heterogeneity of the proposed context-mapping and context-generation modules; modularization enables internal development whilst also lending itself to wider adoption and integration in other systems. Distinct but compatible addressing approaches also arise: Prompt Engineering applies structured tuning in conjunction with LLM context windows, Steering applies augmented contextual prompting to elicit safer, more relevant, and consistent behavior from the model, and Tooling applies a typed interface for code, image, and other media-generation LLM queries, image generation, and API-invocation tasks, permitting and supporting parallel execution.

Table 1 — Symbols used in the derived model

Symbol	Meaning
q	user query
$G = (V, E)$	enterprise knowledge graph

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Symbol	Meaning
H	graph traversal depth / number of hops
B	average branching factor
C	context richness index
R	retrieval relevance score
N	novelty score
D	diversity score
P	personalization score
Gv	governance / safety score
T	total latency
U	final enterprise utility score

3.1. Design of Agent-Centric Architectures

Design principles for context-aware enterprise agents encompass core aspects of modular decomposition, agent interoperability, security-by-design techniques, and provision of evaluation-ready artifacts. Agent-based integration offers architectural flexibility, simplifies user interaction, and scales naturally with enterprise growth. Computing and data usage costs generally remain manageable, with the caveat that extreme latency-critical applications such as customer-facing chatbots must configure agents with synchronous retrieval pipelines.

Incorporating context into agent responses requires careful selection of context sources, modeling of the relevant contextual aspects, and effective integration of context into the response generation process. Supporting context-external factual data and ensuring the novelty of the derived information improves quality and user trust. User queries can be further personalized through the use of building blocks tailored for user and role modeling, as well as domain-specific ontological knowledge. Contextualization of user input is especially important in open-domain settings. Enterprise knowledge bases constituting company-specific ontologies complement task-relevant context by steering the agent's attention toward answer candidates that best reflect the user's query intent.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

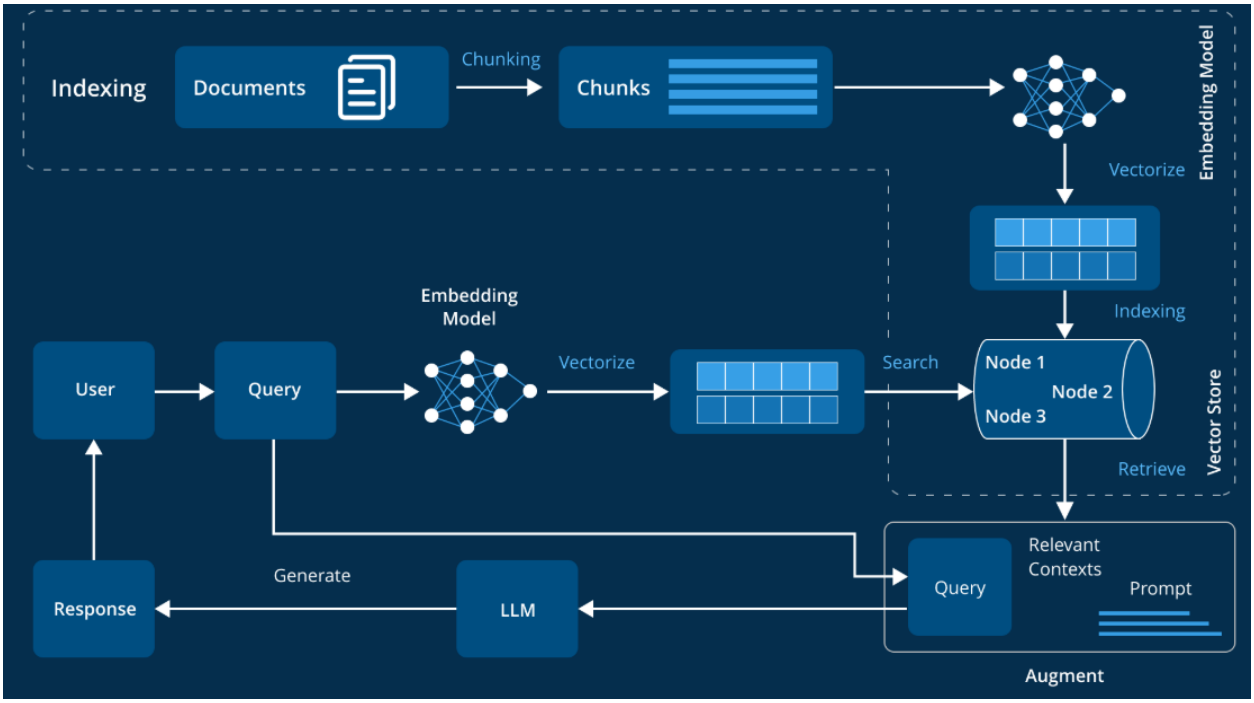


Fig 2: Advanced RAG: Architecture, Techniques, Applications and Use Cases and Development

4. Objectives of the Study

Aiming to improve response relevance, correctness, novelty, and contextual awareness, the proposed architectures are a step towards addressing long-standing requirements of context-aware AI agents that remain unmet by existing large language model (LLM) interfaces and chatbots. Empirical evidence for impact is key in enterprise environments; therefore, the proposal seeks to deliver not only higher but also lower-latency quality responses that can be labeled as safe and correct yet still potentially lead to violations of established guidelines.

Five objectives underpin the development of the contexts tailored for a customer support and service automation use case: better retrieval, improved personalization, richer contextual reasoning, enhanced governance, and enterprise validation that can be measured in practice.

4.1. Goals and Aspirations of the Research

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Goals and aspirations include enhancing retrieval performance, enabling contextual reasoning via knowledge graphs, providing user and role personalization, facilitating automated governance of large language models, and investigating measurable impact on enterprise operations. A set of elegantly designed modular system components for enterprise AI agents underlies these ambitions.

The pursuit of effective conversational interactions between humans and machines is among humankind's oldest aspirations. Early 21st-century advancements in Large Language Models (LLM)—the generative, retrieval-augmented, and multimodal variants in particular—have brought this dream closer than ever. Nevertheless, a broad spectrum of practical business applications remains unrealized. Vital operational domains—customer support, service automation, knowledge management, enterprise search, and much more—remain largely unchanged in their interactions with customers and within organisational silos. Responding quickly and appropriately to real-time questions without guidance from others remains a key frontier among conversational AI tasks—one that real LLMs and current implementation approaches still often struggle to address. Meaningfully addressing these critical limitations of applied enterprise AI agents requires a different kind of emphasis: not on advancing the interaction engine itself, but on its surrounding ecosystem and the essential processes that it supports.

The divergent nature of the challenges and concerns behind fulfilling human-agent dialogue aspirations demands a systemic, agent-centric perspective. AI chat agents and technologies can be analysed and examined as Enabling agents in a Supporting ecosystem. Enterprise Requirements and Concerns define confidence and acceptance demands over the engine and the control processes that keep it safe, secure, and useful. Contextual Knowledge Management via Knowledge Graphs enables the outside information sources and resources to be ever-present, always ready to respond, and directly accessible to aid consideration, assistance, conversation, and guidance with each AI agent's user. The Enterprise Agents Control System governs model conditions and data inputs to the LLM engine while supporting Auto-Governing processes to oversee the safety and security of each AI agent.

These many dimensions of need fuse into a set of goals directing discovery and exploration. Enhanced retrieval quality aims to make otherwise hidden information readily discoverable, directly improving support for human AI agent dialogues. Contextual reasoning capabilities via knowledge graphs seek to help an AI agent assess and determine more complex and non-obvious replies. User and role personalization aspirations call for each model to better match the preferences, style, and behaviours of the user interacting with the agent. Automated

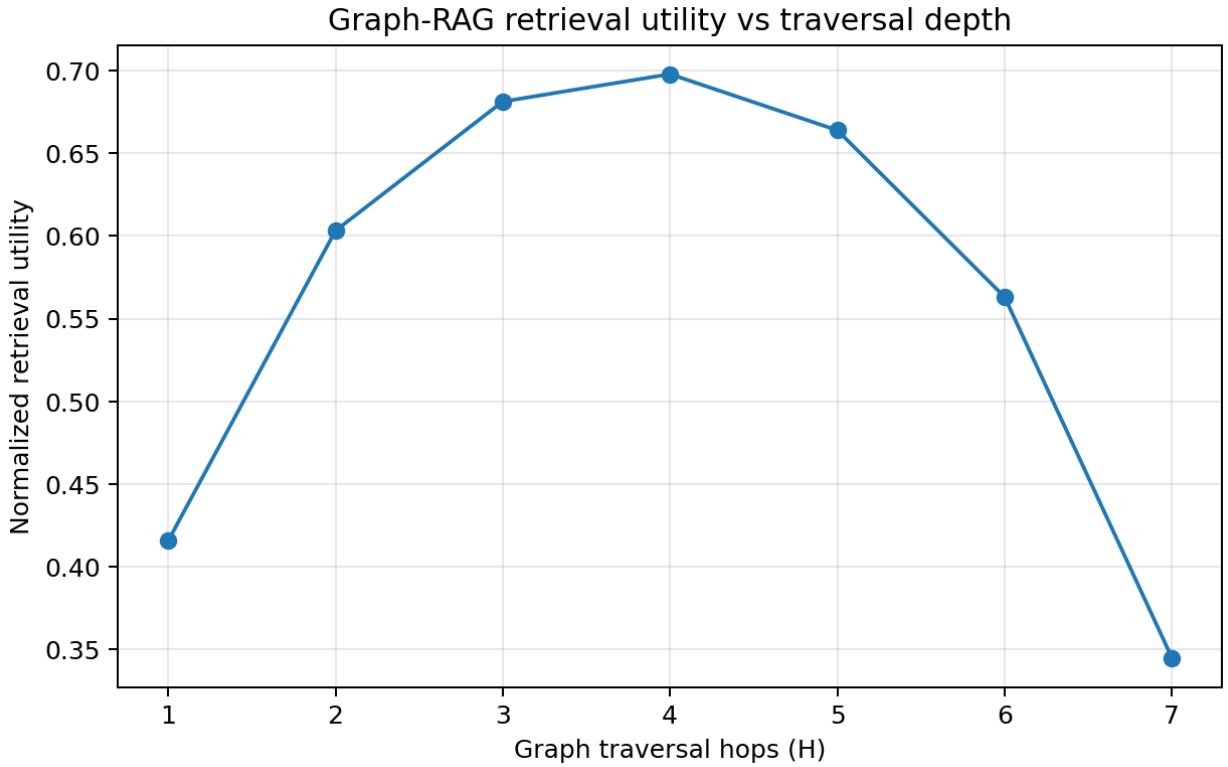
ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

governance provisions make it easy to establish, monitor, and enforce the conditions governing a Large Language Model, eliminating all backdoor access and directly curbing hallucinations and other risks. An interest in measuring enterprise impact ultimately aims to gauge the real business benefits of deploying the new functionality-enabled agents.



Equation 2: Relevance score from embeddings

Step 1: represent query and candidate in embedding space

Let:

$$\mathbf{q} \in \mathbb{R}^d, \qquad \mathbf{x}_i \in \mathbb{R}^d$$

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Step 2: cosine similarity

$$R_i = \cos(\mathbf{q}, \mathbf{x}_i) = \frac{\mathbf{q} \cdot \mathbf{x}_i}{\| \mathbf{q} \| \| \mathbf{x}_i \|}$$

Step 3: range adjustment to $[0, 1]$ if needed

Cosine similarity lies in $[-1,1]$. Convert to $[0,1]$:

$$R_i^{(norm)} = \frac{1 + R_i}{2}$$

5. Research Summary

Architectures for intelligent enterprise AI agents include stateful interaction design, multiplexing for simultaneous multiple interactions, context augmentation by knowledge graph traversal, and user modeling for personalization. Integration patterns indicate that such agents can be built from existing technologies and tools. Collectively, these enhancements improve enterprise RAG capabilities and achieve better results in knowledge management and customer support tasks.

The research advances the design of large language model (LLM)-powered agents by focusing on context and user-specific behavior. Context-aware chat interactions leverage all available knowledge bases and bring in external context at runtime through a novel graph traversal mechanism. User profiles and role models facilitate personalization in domain-specific applications. The evaluation strategy encompasses various domain-independent benchmark tasks and includes an error analysis framework to identify weaknesses and deployment constraints.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

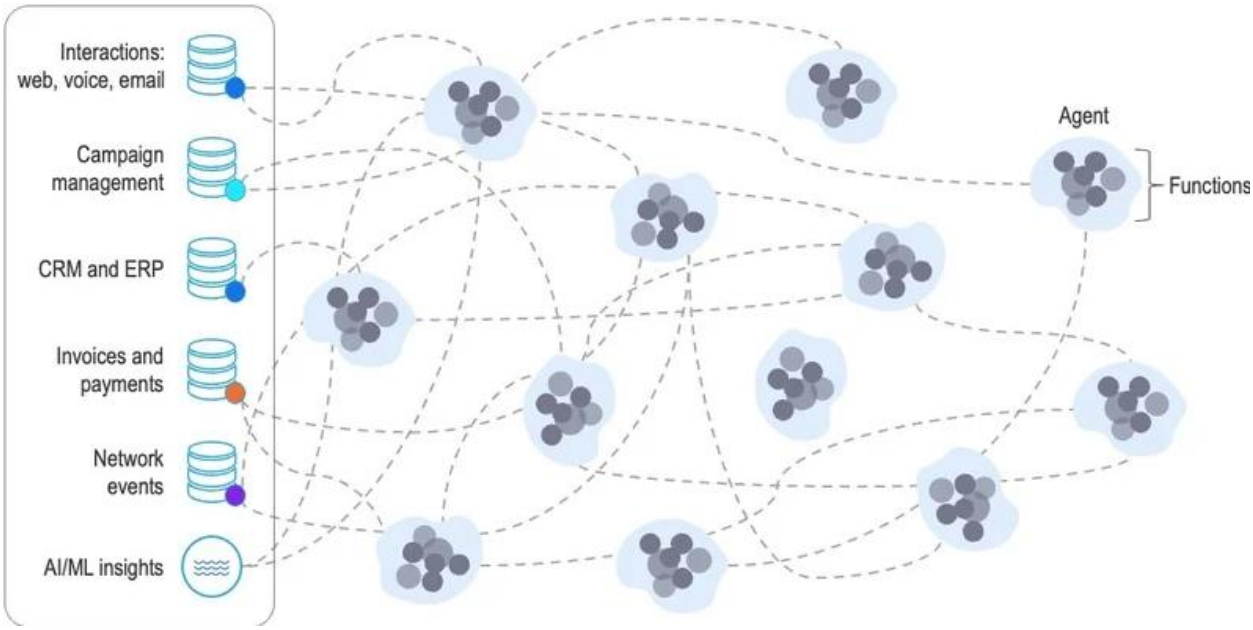


Fig 3: Enterprise RAG

5.1. Architectures for Intelligent Enterprise AI Agents

Intelligent AI agents simplify human-computer interaction by reducing friction across information retrieval, exploration, and task automation activities. In enterprise environments, agent-centric architectures combine Retrieval-Augmented Generation (RAG) models with context management and data pipelines to facilitate interactions with business applications, products, and services. Such systems retrieve relevant information from specialized databases while taking contextual elements—such as user identity, role, location, and conversation history—into account. A proven use-case family leverages language models for decision-support in contact centres by looking up facts to answer questions, proposing responses to incoming messages based on limited contextual history, and escalating to human agents only when necessary.

The resulting architecture integrates Graph-Enhanced RAG candidates with contextual user and role modeling, customized domain taxonomies, and context-sensitive prompt engineering. Domain-specific versioned knowledge graphs capture relationships between entities of business importance—such a product and marketing assets—and support probabilistic tavarsal

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

methods to improve breadth and relevance of information retrieval operations. The effectiveness of context-aware agents can be gauged through a task-centric benchmark suite tailored for intelligent interaction and management of products, platforms, services, and knowledge across the enterprise.

6. Architectures for Context-Aware Enterprise AI Agents

The architecture of agent-centric systems is defined by a modular decomposition supporting independently operable components. Signals flowing through the system at runtime interconnect these components, forming a dynamic and distributed workflow. Evidence in the form of supporting context, supplemental tools, and additional information sources can be seamlessly integrated into the flow. Architectures that deliver context-aware enterprise agents combine multiple providing axes: graph-based approaches enhance specialized retrieval processing, while LLM-based components could serve as retrieval endpoints, task tools, or plug-in associations.

An agent-centric system orchestrates the interplay of the components proposed for improved contextual augmentation, knowledge and tool invocation, and LLM output enhancement. Figure 6-1 presents the overall architecture, with transactional context passing upwards through the user definition and processing layers. Supporting knowledge may be drawn from a number of actively updated and specialized sources that are structured to enable the optimization of task-class-dependent retrieval. The provision of additional non-textual tools would also reside within supporting breadth, facilitating more elaborate transactions such as composite file generation or execution in non-LLM-driven containers.

6.1. Agent-Centric System Architecture

Agent-centric system architecture fulfills diverse roles for intelligent enterprise AI agents. These roles, such as user-facing chatbots, internal service automators, and retrieval-enhanced large language models (LLMs), exhibit differing interaction patterns with retrieval and context-modelling subsystems. Although these variations imply an agent-centric design that factors in distinct components, rigorous implementation ultimately relies on engineering quality and the system’s articulation into interoperating modules.

The proposed architecture decomposes a context-aware enterprise AI agent into user-facing and internal components. Internal components service one or multiple user-facing agents without direct user interaction. Additional interaction partners, such as knowledge bases,

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

databases, and external APIs, complete the system picture. Data-flows between all operational units support the routing of information and maintain the coherence of the data-producing and interaction-supporting pipelines.

6.2. Contextual Knowledge Management

Contextual knowledge management encompasses the creation, maintenance, and use of contextual knowledge. This includes the management of various types of knowledge graphs and facilities for knowledge management, used to create and oversee domain ontologies and other taxonomies. These play important roles in guiding the content of AI-generated responses and are the preferred means for storing and making available structured knowledge sources that support fresh knowledge retrieval modules, especially those used to model user context and those that maintain other types of contextual knowledge.

Enterprise contexts often evolve rapidly, and enabling the effective use of contextual knowledge requires mechanisms to keep knowledge graphs, domain ontologies, and contextual models up to date. Moreover, a disconnect tends to exist between the ontologies and vocabularies used in enterprise applications and the specialized ones used by language models, especially when handling enterprise technical queries. Regular alignment of domain ontologies with the enterprise taxonomy, together with update mechanisms and consistency checking, are therefore critical for effective prediction and completion of usage-based context-sensitive prompts.

Table 2 — Derived qualitative

Dimension	Standard RAG	Graph-RAG / Context-Aware Agent	Why the article suggests this
Relevance	Medium	High	graph traversal + domain ontology + task-specific retrieval
Novelty	Medium	High	candidate scoring can include novelty and provenance
Personalization	Low	High	user model + role model + context profile
Governance	Medium	High	policies, controlled tools, ontology-guided retrieval

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Dimension	Standard RAG	Graph-RAG / Context-Aware Agent	Why the article suggests this
Latency efficiency	High for simple cases	Medium	richer retrieval and tooling add overhead

7. Graph-Enhanced Retrieval Modules

Graph-RAG for context-aware Retrieval-Augmented Generation relies on a graph-enhanced retrieval pipeline. The quality of the retrieved candidates profoundly influences the overall performance of the architecture. When reasoning about a specific context during RAG, a context-aware system should ideally focus all its cognitive abilities on that context, securing the best available quality for the task involved. In this light, the two main aspects of the candidate retrieval process are the module used for candidate discovery and the traversal strategy employed to explore the graph for candidate identification. Each of these aspects is captured in a dedicated subsection.

Candidate discovery involves the selection of a retrieval approach and scoring function that is coherent with the data types and qualities being exploited for RAG. The availability of heterogeneous sources allows for a richer candidate pool, but it also raises exposure risk and irrelevant support along the generated completions. Leveraging such a diversity therefore hinges on the ability to score candidate novelty and relevance jointly with the common RAG ranking. A set of heterogeneous graph embedding strategies, tuned to both the specific data type characteristics and a common schema, should aid in producing a generalizing representation space and allow for a heterogeneous candidate pool. Candidate scoring must combine novelty measurements and relevance with respect to the LLM query inspired by the available context.

The traversal method deals with the search patterns used to retrieve candidates during a specific context. Suitable operations can ensure that the search effort is focused on the neighborhood of the currently active context. This not only improves latency, as the queries scale better with context complexity, but also enhances the results by prioritizing candidates that describe the same context in different sources.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

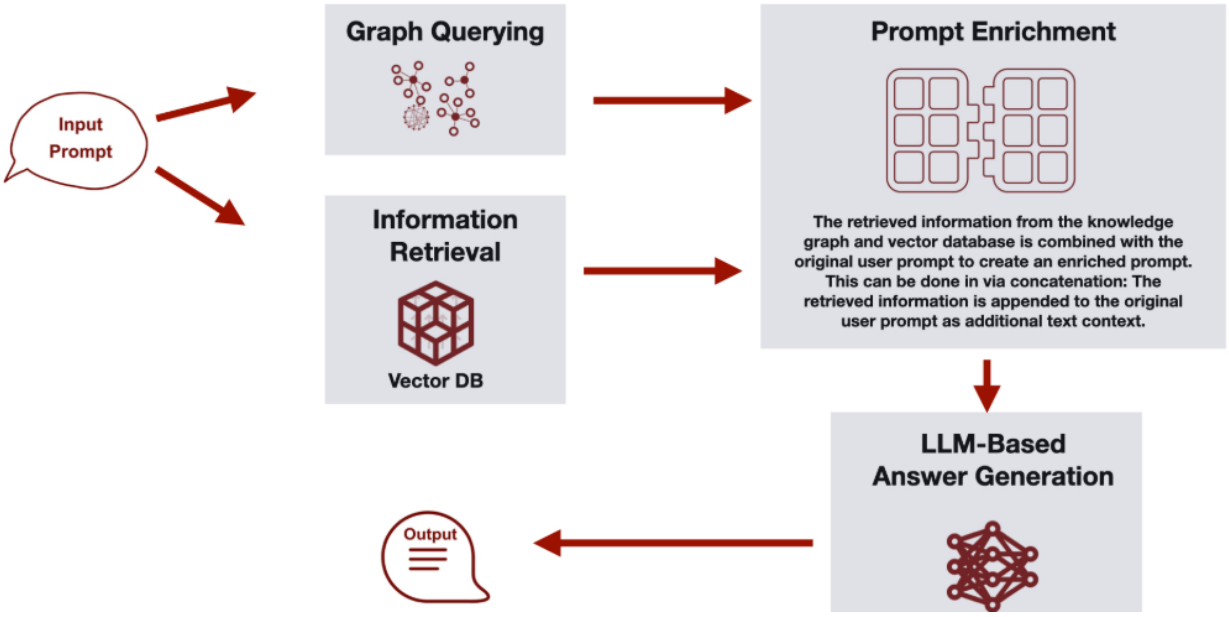


Fig 4: GraphRAG: Design Patterns, Challenges

7.1. Graph-Based Indexing and Embedding Strategies

A key requirement to realize the Graph-RAG architecture is to adapt existing retrieval methods for knowledge-graph-augmented LLM contexts. The aim is to identify and score relevant documents, knowledge-graph subgraphs, and domain-specific candidates for user queries in a sufficiently low-latency manner. To utilize heterogeneous data stored in different formats, embedding models capturing the integrity of documents and graphs need to be trained on semantically comparable schemas. An additional challenge is to enrich the encoded information from the resources being traversed. Common graph-based retrieval methods such as graph traversal and subgraph matching have so far concentrated on the semantic content of graph edges and nodes, but the graph can offer much more (e.g., support for diversity and novelty) if quality attributes for the semantics stored therein are defined and exploited.

Latent-knowledge discovery and distribution based on information-theoretic measures could also be valuable for knowledge-graph-augmented LLM contexts. Unfortunately, standard RAG setups only capture these information-theoretic properties when proper candidates are retrieved from the knowledge graph. The prioritization of domains, classes, and individual

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

entities during candidate selection is hardly ever considered yet can strongly affect novelty, diversity, and completeness. Well-structured domains and well-maintained ontologies offer rich domain knowledge, and probabilistic or heuristic models can help drive candidate selection even in less-controlled environments.

7.2. Traversal, Subgraph Matching, and Candidate Retrieval

The candidate retrieval phase begins with processing a user input query and identifying the retrieval task category and search target. A query decomposition step supports command-type queries; for other query types, the retrieval pipeline first aims to identify the most relevant subgraph. The target subgraph serves as the primary search space, guiding embedding alignment, candidate matching, and traversal operations. If a dedicated task-specific search target (e.g., a specific graph-based knowledge base) is provided in the query context, subgraph selection is bypassed.

Graph traversal or subgraph matching operations follow, making use of the high-level retrieval goal to identify a smaller subset of candidate entities or graph paths. Entities or paths matching the user query at the most concrete level generate candidates for ranking—if the query specifies a question, candidate answers are classified, ranked, and optionally filtered for novelty—or are forwarded to an appropriate external information source. Quality, novelty, and relevance evaluations iteratively filter the remaining candidates at the semantic level, and context-injected passages augment the final response before it is returned to the user.

8. LLM Integration Strategies

Thoughtful prompt engineering and flexible plugin architectures are explored for integrating large language models (LLMs) into context-aware agents based on principled graph-enhanced retrieval-augmented generation. First, special attention is given to the architecture of LLM prompts to ensure the models emply all available contextual clues, produce safe completions, and provide interpretable results. Second, a custom plugin interface is introduced to facilitate the addition of arbitrary tools and other context-awareness strategies without necessitating any architectural refactoring.

Prompt engineering of large language models (LLMs) and retrieval-augmented generation (RAG) interventions play a critical role in maximizing the useful influence of context on agent behavior. During prompt construction, the effective context window of LLMs is filled

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

with context-relevant information, including instructions and safety guidelines for sensitive tasks.

Graph-enhanced RAG agents benefit particularly from prompt engineering, as they frequently return lengthy passages retrieved from dedicated data sources, which can be integrated into the context window to help guide LLM behavior. The presence of retrieval results also permits the context-aware presentation of completions in style or format that suit the target audience. To illustrate this, every RAG result is considered an individual node in the generated LLM prompt, allowing the model to sense that the RAG component advises a new task or question. If a RAG sample has a low novelty score, that node remains in the prompt context window but is visually distinct (e.g., smaller font) to highlight redundancy.

The addition of other tools and utilities can improve agent behavior quality and diversity. Recent developments in LLMs showing some reasoning capabilities when asked to complete code fragments yield interesting opportunities when the LLMs can internally provide completion solutions for other integrated tools, enabling their concurrent use. With dedicated function headers, arguments, tool signatures (input and output examples), and appropriate definitions of three guiding completion types for tools (in addition to standard completion-type prompt messages), the functional completion type can be added to any LLM integrated into the workflow.

Equation 3: Novelty score

Let selected candidates so far be:

$$\mathcal{S} = \{x_1, \dots, x_k\}$$

Step 1: define redundancy of x_i against already chosen items

$$\text{Redundancy}(x_i) = \max_{x_j \in \mathcal{S}} \cos\left(\mathbf{x}_i, \mathbf{x}_j\right)$$

Step 2: novelty is inverse redundancy

$$N_i = 1 - \max_{x_j \in \mathcal{S}} \cos\left(\mathbf{x}_i, \mathbf{x}_j\right)$$

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Step 3: interpretation

- if x_i is very similar to an already chosen result, novelty is small
- if x_i contributes new information, novelty is large

8.1. Prompt Engineering for Graph-RAG

Prompt engineering for Graph-Enhanced RAG (Graph-RAG) systems involves configuring prompts and tool definitions to produce an agent's act in a way that exploits RAG's contextual capability while leveraging Graph-RAG's broader contextual information and reasoning capability. Parameters that a researcher or developer needs to focus on are (1) the composition of the context window fed to the LLM, (2) additional information and instructions provided to the LLM as part of the prompt, and (3) additional instructions provided to the LLM for safe operation of the agent when using LLMs such as OpenAI's ChatGPT or Anthropic's Claude that have safeguards against producing harmful content.

A typical context window consists of the subgraph for the current user or role (if present) obtained using simple user-role-based access control checks, persistent user-role-specific preferences about chat completion, and/or recent conversation history. An additional set of instructions using a carefully-designed system prompt may be provided at the start of each agent act to explain the purpose of the agent and other constant aspects of its operation. In every act, a flexible user-provided prompt in the user-role-specific profile may be supplied to provide chat guidance or invoke specific features, including narrowing the scope of generation, enabling domain-specific jargon or style, and so on. A flexible novelty-scope prompt may also contain a prompt template and token replacement map to produce additional content-specific drilling instructions. Paraphrase-level reliability is achieved by using an effective paraphrasing tool in the speech synthesis phase.

8.2. Tooling and Plugin Architectures

The tooling and plugin architectures of context-aware enterprise agents define a set of application programming interfaces that enable the agent to enhance its capabilities when required by the task at hand. These interfaces expose the necessary functionality and data access for users, systems, or other agents to perform specific actions on behalf of the assisting AI. Examples include search tools for structured data, specialized sources of information such as

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

databases or knowledge graphs, and access to third-party services such as transport or booking services.

Since tooling requires task-specific functional capabilities that go beyond the general-purpose strength of large language models (LLMs), this means that it is possible to be more permissive in the interaction with the actor, which can act almost without constraints when supported through API calls. The key consideration here is how the integration should be structured: options include sequential execution (with the model triggering a tool call, waiting for the results, and executing next actions) or parallel calls (with multiple tools being triggered at once, so the latency overhead incurred by triggering the tool is reduced).

9. Context Modeling and Personalization

Context modeling and personalization comprise techniques for capturing and utilizing information on user identities, roles, and preferences, domain knowledge, and organizational taxonomies. Agents based on Graph-Enhanced Retrieval-Augmented Generation (Graph-RAG) and context-aware Large Language Models (LLMs) benefit from context information in various ways. User modeling enables personalization of responses, making interactions more relevant, engaging, and secure. Domain ontologies and taxonomies encodes specific knowledge about the application domain—for example, taxonomies of part, product, or service types—allowing agents to converse in terminology specific to that domain and improving the quality of knowledge retrieval when subsections of the ontology are employed as context. Furthermore, an enterprise-wide ontology or taxonomy ensures consistency across differing agent instances.

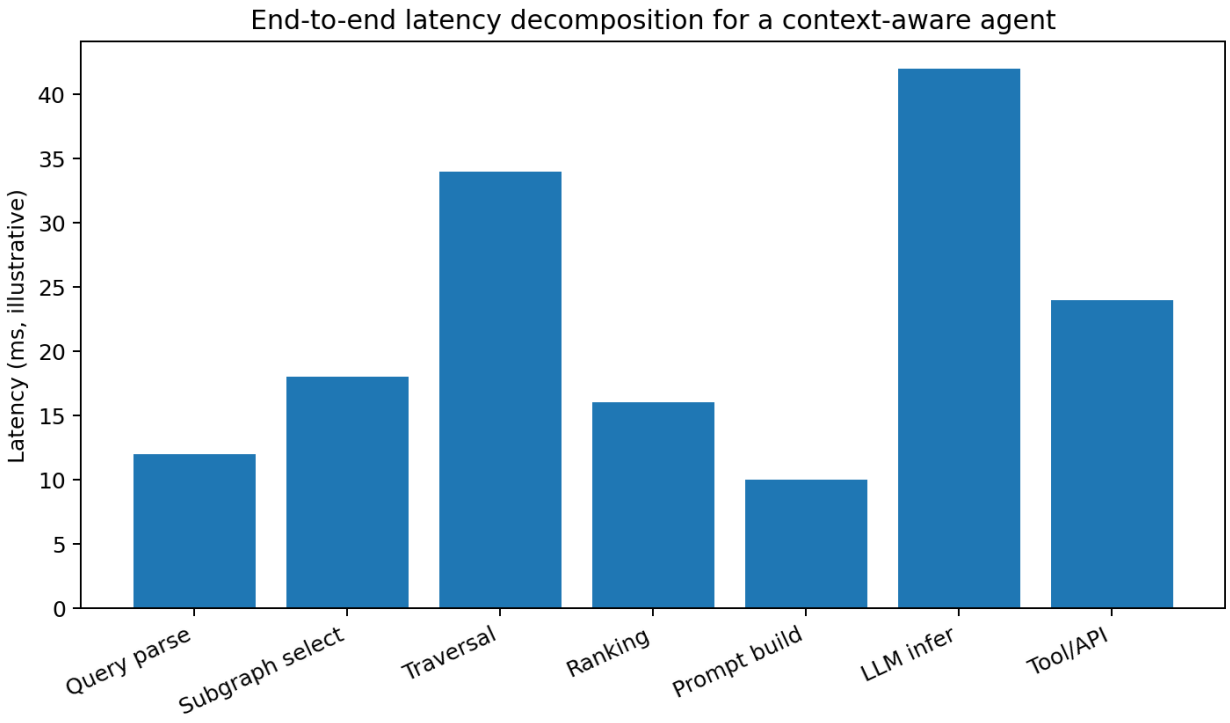
Incorporating context information raises privacy questions: any service that collects and stores a user profile must protect the stored information from unwanted access. To mitigate this risk, context information—especially user preferences—might be computed and maintained by the user or by a decoupled User Preference Service hosted within a private domain. Modelling User Profiles, Role Profiles, Domain Ontologies and Taxonomies, and Context Management. No additional context needs to be provided for users that have not logged in or established a working session with an agent. A User Profile is created and stored for such users for the duration of the service session, after which the profile is discarded. User preferences must be safely discarded once a user logs off or completes an interaction in a responsible manner since these can include personally identifiable information that should not be shared with others.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22



9.1. User and Role Modeling

User and role modeling comprises the specification and management of digital profiles for individual users, encompassing office workers, decision-makers, and customers, as well as collective profiles for functional roles (marketing, engineering, customer support). Users can express their interests, preferences, and restrictions about workplace processes and tools, enabling the agent to act in a more personal and relevant manner. For example, it can consider user preferences for human-like dialog (friendly/very formal) or level of detail (high-level summary/in-depth technical analysis) when answering questions or producing documents. Users are motivated to share such information if privacy issues are handled appropriately and if it helps to improve the quality of the responses. Privacy and safety are main concerns, as this information may reveal sensitive details about users.

Collective profiles store high-level information about functional roles, capturing common preferences and restrictions. They also serve as a support for enterprise-level customer

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

interaction services. Customer profiles provide information that allows a more personal interaction in customer service scenarios, offering agents the ability to adapt tones of voice or responses according to the type of customer—experience level, VIP status, etc. Domain ontologies and associated taxonomies play a central role to help an agent provide focused and proper answers according to the user's inquiry domain. These knowledge sources should be aligned and maintained up to date with the enterprise information system: Human Resources Management, Corporate Taxonomy, website, etc. Updates and modifications must be performed regularly, new domains added when new products/services are launched, and a consistency-checking mechanism should ensure that all ontological concepts are correctly organized.

9.2. Domain Ontologies and Taxonomies

A critical component for improving contextual knowledge is user and role modeling, aiming to define and refine user profiles and preferences while respecting privacy. Contextualization generation may be simplified and personalized using domain ontologies and taxonomies that maximize coherence with the enterprise taxonomy, are readily updatable, and can be tested for redundancy and consistency.

Domain ontologies define concepts and relationships in confined environments, whereas taxonomies consist of labeled hierarchies. These structural specifications significantly lower the cost and effort of contextualization by restricting the range of possibilities for all generations. However, roles may be extremely diverse and specific to individual users. Taxonomies should thus be aligned with the enterprise knowledge graph taxonomy and regularly checked for redundancy and consistency, while user collections of other-context settings and preferences should remain as unconstrained as possible. Enterprise safety policies should dictate additional constraints on potentially harmful generations and usages.

Whitelisting or blacklisting infrastructure may guide or restrict generations using address- and element-specific prompts. Tools and external plug-ins accessible from the context may be automatically adapted to individual users by enabling, disabling, or modifying tool-specific prompting attributes based on user profiles and other-context submissions.

10. Evaluation Methodologies

Evaluation plays a crucial role in developing context-aware agents. Indeed, automated evaluation forms a prerequisite for training and fine-tuning reliable yet performant enterprise AI agents. Such agents must derive context from user inputs or inputs retrieved via context-aware

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

search and filtering techniques. Consequently, the primary evaluation benchmarks for such contexts are correctness and relevance. The proposed userTaskBenchmark should cover interaction workflows where users seek information, guidance, or support from agents or chatbots. The chat data should be partitioned into training, evaluation, and testing sets. Customers may also prefer to search knowledge bases or query databases. Automated question-processing workflows are increasingly being adopted by data-intensive domains such as education, healthcare, and customer support, where customers expect responses within shorter time windows. The response latency of agents and chatbots emerges as a second important metric.

Precision and recall estimators at the corpus level during the deployment phase can generate insights on the effectiveness of index-based search strategies. Latency emerges as a third important consideration, especially for traversals requiring multiple hops for retrieving relevant information. Adding tooling, plugin, and browsing capabilities to LLMs increases their functionality beyond pure text completion yet introduces latency overheads that many users would like to minimize. Section elaborates the construction of a task-focused benchmark suite for context-aware agents engaged in information-seeking workflows. The objective is to create commonsense-like datasets in the public domain that enable easy validation and reproducibility of development efforts and promote the training and fine-tuning of high-accuracy, low-error-rate, and low-latency agents.

Error analysis aids in broadly understanding the reliable and unreliable domains for each deployed agent type and performing failure-mode analysis. A combination of gene-ralized precision and generalized recall metrics can be employed to quantify the accuracy, relevance, and novelty of candidate retrieval strategies. Interpretability and response safety thus emerge as important aspects to consider during the analysis of manually curated agent responses.

Equation 4: Context fitness score

Let:

- U_i = match with user profile
- Ro_i = match with role profile
- Do_i = match with domain ontology

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

- H_i = match with recent interaction history

Step 1: additive composition

$$C_i = w_u U_i + w_r Ro_i + w_d Do_i + w_h H_i$$

Step 2: weight normalization

$$w_u + w_r + w_d + w_h = 1, \qquad w_u, w_r, w_d, w_h \geq 0$$

Step 3: if each component is cosine similarity

$$U_i = \cos(\mathbf{x}_i, \mathbf{u}), \quad Ro_i = \cos(\mathbf{x}_i, \mathbf{r}), \quad Do_i = \cos(\mathbf{x}_i, \mathbf{d}), \quad H_i = \cos(\mathbf{x}_i, \mathbf{h})$$

Then:

$$C_i = w_u \cos(\mathbf{x}_i, \mathbf{u}) + w_r \cos(\mathbf{x}_i, \mathbf{r}) + w_d \cos(\mathbf{x}_i, \mathbf{d}) + w_h \cos(\mathbf{x}_i, \mathbf{h})$$

10.1. Benchmark Suites for Context-Aware Agents

Context-aware agents address the failure of traditional chat-oriented dialogue systems to offer reliable real-time answers in high-stakes environments. Chat-based RAG agents analyze user prompts, identify relevant knowledge, and formulate complete replies using a generative language model. However, the lack of task-directed objectives makes chat conversation benchmarking less meaningful than established relevance-grounded frameworks commonly employed in information retrieval, knowledge management, or classification tasks.

To fill this gap, a set of context-aware agent benchmark tasks covering customer support, service automation, enterprise knowledge management, and search has been defined. Automatic gold-standard labels enhance benchmark quality by enabling precise evaluation from a retrieval-centric perspective. The resulting data, currently used to assess a context-aware agent instantiation and supported by two baseline solutions, will help improve context-aware RAG agents, provide insights into scaling concerns, and monitor novel failure modes.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

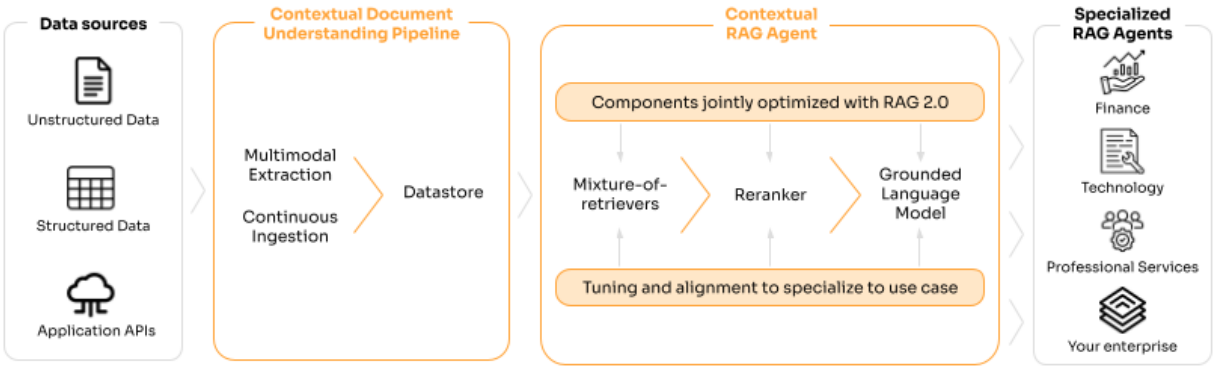


Fig 5: Benchmarking Contextual RAG Agents

10.2. Task-Centric Metrics and Error Analysis

Task-centric evaluation metrics augment traditional accuracy measures—precision and recall—with additional measures of latency, interpretability, and semantic errors. Latency encompasses response time for generation and retrieval components, as well as end-to-end latency from query input to final answer presentation. Latency is especially relevant for automated customer support agents, where response times significantly affect customer satisfaction and agent effectiveness. Interpretability assesses the degree to which task inputs can be understood by human readers without special training. Greater interpretability supports human users when following agent-supplied instructions or performing other complex tasks. Errors relate to failures in communication or reasoning, and include hallucinations, contradictory statements, reference errors, redundant information, and incorrect instructions. Semantic errors may be especially significant in end-user-facing agents and task environments characterized by limited user knowledge.

Task-centric evaluation avoids direct comparison of systems with different architectural components by defining baselines for each component pairing. Regression suites support error classification, while new tasks can be executed with preserves training and evaluation splits but with the Task class modified to support the new task type. Chat datasets can further be divided into individual user messages to form datasets for dedicated RAG-powered customer support or service automation agents. Splits must guarantee that predictive context windows for benchmark heuristics can be filled by the non-task-centric hybrid.

11. Industrial Applications and Use Cases

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Intelligent agents that proactively assist with internal and external queries are increasingly used by enterprises to improve customer service experiences, optimize resources, and lower operating costs. These tasks can be fulfilled by dedicated customer service agents or by more generic function agents whose responsibilities, capabilities, and areas of authority are driven by user roles and contextual prompts. The former typically engage in chat conversations, retrieving relevant information from support knowledge bases and addressing complex queries by escalating to human agents. The latter assist users across multiple functions and interact across different modalities. External user interactions are similar in nature to generic chatbots, requiring a mechanism for retrieving external references and product- or service-related information. The architecture is designed to facilitate such task automation.

In an Intelligent Enterprise Search and Knowledge Management setup, complex Indexing workflows are required for scalable enterprise-specific search engines. Existing solutions offer standalone capabilities; for example, some focus on knowledge management but do not allow chat workflows with support retrieval of non-knowledge-management information. The focus here is to define an agent-centric architecture for such contexts, along with both dedicated and generic function modules. These operational details guide the instantiation of two use cases, intelligent customer automation workflows supporting a decision tree to manage escalation of complex queries and an enterprise search module helping information workers discover and share information.

11.1. Customer Support and Service Automation

Large organizations frequently face pressure to support customers across multiple products with varied or highly expert content and to automatically assist customers with none-to-minimal agent intervention. This goal can often be effectively approached as a natural language understanding (NLU) or generation (NLG) task. During a chat, the current subject is detected from either the chat content or metadata flags. Targeted retrieval is then performed against knowledge bases—support documents, product manuals, outcome reports, and escalated ticket descriptions. If the agent detects high confidence or an obvious answer, it responds; otherwise, it returns “I don’t know,” with the chat history so far concatenated to the prompt as context.

While (N)LU directly detects the next probable statement, it can also be considered a super-agent, where a number of retrievers, assistants, and scripted chat assertions evaluate the state and content in the live chat. As with every Chat service, downsides can occur, even when somewhat intelligent chat messages are shown: the response might be inappropriate or false. In this context, agents can even create incorrect NLU/NLG mappings when trained using traditional

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

supervised learning methods at surface probability, influenced by their prior probability, and hence a mixture model is possible. These interactions potentially reduce user agency more than plain FAQs, since they now produce limited-domain sentences.

Table 3 — Illustrative latency breakdown used for Graph 2

Component	Latency (ms)
Query parsing	12
Subgraph selection	18
Graph traversal	34
Candidate ranking	16
Prompt construction	10
LLM inference	42
Tool/API call overhead	24
Total	**156**

11.2. Knowledge Management and Enterprise Search

A critical large-language-model (LLM) use case involves knowledge management and enterprise search, where timely and accurate question answering is essential for maintaining customer satisfaction and service quality. To address this need by leveraging knowledge graphs and enabling the incorporation of enterprise-specific data sources, the design integrates data conditioning and augmentation modules. These modules address all major aspects of enterprise search, allowing custom queries that adhere to an organization's established policies. Rather than simply paraphrasing established codified knowledge, the system can also traverse knowledge graphs, ensuring that at least some of the answering evidence comes from a source that has been sufficiently validated for answering a particular piece of information.

These capabilities not only improve safety and assurance but also support enterprise-specific functions, such as searching for an accurate price of a product. An enterprise may have information that comes from third-party vendors, and specifically agreed prices may change periodically. Consequently, any enterprise installation must allow continuous updates to the underlying data. Besides these advantages, the inclusion of enterprise context and a knowledge-graph-based retrieval mechanism allows for generic Q&A and free-form voice- or text-based

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

searches, broadening the scope. A dedicated search engine, augmented with a voice interface, thus allows users to find relevant sections of huge document sets and large websites through semantic embedding-matching techniques.

12. Challenges, Risks, and Ethical Considerations

Scalability, Latency, Cost, and Operational Constraints

Graph-enhanced graph retrieval-augmented generation and contextual LLM architectures improve the quality, performance, and governance of intelligent enterprise AI agents. A modular and open design supports an agent-centric system architecture, enabling an ecosystem of context-aware enterprise AI agents tailored to a wide range of functional areas and business applications. However, the application of newly proposed Graph-RAG architectures is accompanied by challenges and risks associated with scalability, operational costs, and the use of external data for initialization and augmentation.

The primary risk associated with the architecture at scale is the inevitable increase in operational latency associated with additional context modeling, complex graph-enhanced retrieval processes, and frequent integration of graph-enhanced context into LLM prompting. Architectural patterns, design choices, and technology selection can mitigate some of this latency. For example, leveraging LLM tooling or plugin architectures can help parallelize context-aware functionality to reduce overall end-to-end latency. Nevertheless, the problem remains a fundamental one with no silver-bullet solution.

Data Quality, Graph Reliability, and Reasoning Certainty

The reliability of contextual reasoning also remains a risk as long as the underlying retrieval systems cannot be entrusted to provide correct, complete, and timely context. Particular care is therefore required with respect to the quality of any external data sources—whether happy or unhappy data quality engineering. Provenance tagging prepares the data for consistency checks using general-purpose prediction models familiar to every major LLM developer and requires only minor domain-specific adaptation to become operational. Consistency-checking procedures can readily be embedded into the Enterprise Taxonomy Ontology Spaces and tested on a reasonable subset of the knowledge graph.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Equation 5: Graph traversal size and latency

Assume:

- branching factor B
- traversal depth H

Step 1: nodes visited at each hop

At hop 1:

$$\boldsymbol{B}$$

At hop **2:**

$$B^2$$

At hop 3:

$$B^3$$

So through hop H, total explored nodes:

$$N(H) = \sum_{h=1}^{H} B^h$$

Step 2: geometric series expansion

$$N(H) = B + B^2 + \cdots + B^H$$

For $B \neq 1$:

$$N(H) = \frac{B(B^H - 1)}{B - 1}$$

Equivalent form:

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

$$N(H) = \frac{B^{H+1} - B}{B - 1}$$

Step 3: latency proportional to explored nodes

If each visited node costs c time units and base overhead is t_0, then:

$$T_{\text{trav}}(H) = t_0 + c\,N(H)$$

Substitute $N(H)$:

$$T_{\text{trav}}(H) = t_0 + c\left(\frac{B^{H+1} - B}{B - 1}\right)$$

Step 4: with pruning factor $p \in (0, 1]$

Effective branching factor:

$$B_{\text{eff}} = pB$$

Then:

$$T_{\text{trav}}(H) = t_0 + c\left(\frac{(pB)^{H+1} - pB}{pB - 1}\right)$$

12.1. Scalability and Latency Trade-offs

Providing context-aware responses to open-ended queries typically involves multiple processes, which can increase throughput latency relative to dedicated components. Contextual support for simple end-user queries may be implemented by embedding context into prompt engineering, thereby enabling a single-pass architecture. Researching several challenge-oriented dialogue tasks makes architectural trade-offs evident. In particular, the KORE-BASED-NAVIGATION task demands low-latency processing Copyright © 2023 Safar Shahraki Chahak, Ahmad Kharazmi, Soufiane Rachidi, Matthias S. S. Kauffmann, and Volker Tresp 12 while enabling rich contextual support. During customer interactions, LLMs may rely on simpler plugin mechanisms, augmenting sensitivity, safety, and interpretability, without compromising LLM strengths.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Training and operating contextual agents is materially more complex and costly than hard-coding functionality. Evaluation-ready yet open-ended RAG modules are excessive for common enterprise tasks. However, their research-value trade-off may be attractive in use cases with rich back-end, user, or interaction complexity, where multistage RAG latency is less critical. Formal designs inherently consider such trade-offs, empowering focused research efforts.

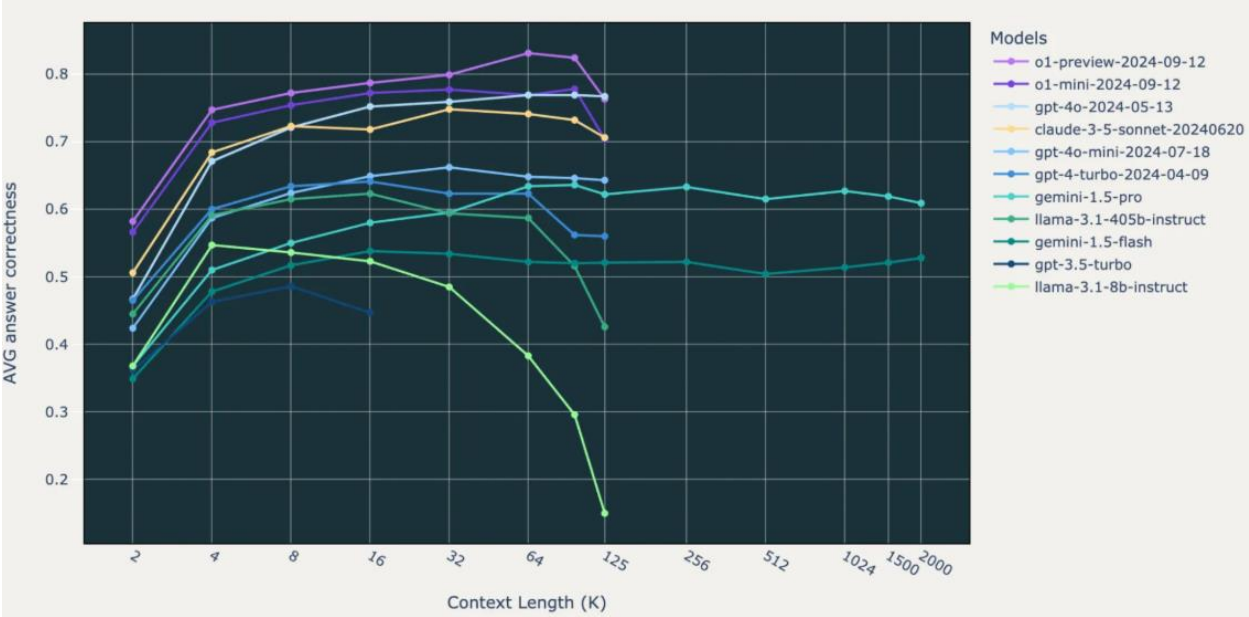


Fig 6: Long Context RAG performance of LLMs

12.2. Data Quality and Graph Reliability

The reliability of the knowledge base and graph plays a crucial role in enterprise agents' safe operation. Data quality and correctness have a direct impact on the quality of index and query responses and, consequently, on the expected agent behavior. Provenance information supports data trustworthiness by documenting and processing sources, nature, and owners. Outdated or incorrect information can harm the AI agent's performance and reputation, prompting the need for continuous maintenance and dependable source validation. Graph and knowledge updates and quality checks can reduce such risks.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Classical consistency checking of knowledge graphs and general-purpose reasoning over ontological schemas may be insufficient because enterprise knowledge representations are subject to rapid modification by product or service evolution, personnel changes, and AI-assisted automatic addition of novel entities or relations. Updates may introduce inconsistency but could remain untriggered by common user requests. Syntactical and semantic validation may complement consistency checking, focusing on local structures preceding or following major enterprise events. A data-oriented monitoring system safeguarding maintenance routines and specialized cleanup tools could complement regular checks, focusing on potentially risky areas.

13. Results

Operations, knowledge flow, and interaction patterns comprise an agent-centric system architecture. Role-based knowledge management deploys specialized knowledge graphs, domain ontologies, and task taxonomies to support effective knowledge retrieval, dialogue generation, and enterprise search.

For enterprise applications that demand context understanding and personalization, the proposed Graph-Enhanced RAG architecture and context modeling capabilities significantly enhance retrieval quality, broaden contextual integration, and enable both personalization and privacy-preserving user support. The anticipated improvements extend to other context-aware agent designs supported by similar architecture concepts.

Generative AI systems for use-cases that require context-aware functionality benefit from agents that model role-based context. With the introduction of context modeling capabilities to the underlying retrieval engines, user context profiles, domain representation, and a task-specific taxonomy, the availability of context-appropriate task guidance information in the generated output aligns better with user needs and expectations. The enhanced context knowledge enables retrieval modules to search for task-related content more effectively and enrich the generative response with an appropriate grasp of the dialogue or answer context, thereby improving consistency and coherence.

Table 4 — Example utility values vs hop depth used for Graph 1

Hop depth H	Relevance term	Novelty term	Latency penalty	Utility U_H
1	0.573	0.347	0.010	0.410
2	0.817	0.548	0.030	0.580

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Hop depth H	Relevance term	Novelty term	Latency penalty	Utility U_H
3	0.922	0.663	0.070	0.637
4	0.967	0.730	0.150	0.626
5	0.986	0.768	0.310	0.580
6	0.994	0.790	0.630	0.434
7	0.998	0.802	1.270	0.084

14. Conclusion

The presented architecture enables the development of Graph-Enhanced RAG usage scenarios (e.g., customer support automation, conversational knowledge retrieval) and shows how enterprise AI agents capable of personalization transform knowledge management and enterprise search. With these capabilities, the Graph-Enhanced RAG approach moves toward industry impact through the deployment of additional solution components (e.g., graph-based indexing strategies and versioned knowledge graphs) and specific evaluation methodologies targeting context-aware agents. The first of these devoted benchmark suites leverages customer-support scenarios for exploration and monitoring, ensuring that model responses remain precise, relevant, and aligned with company values.

Graph-Enhanced RAG architectures prepare LLMs for high-quality Internet-scale RAG and human conversations, enhancing response relevance in domain conversations. The reactions of customers and product users to Azure OpenAI Service ChatGPT and Claude promises well-structured completion responses enabling deployment readiness and usage confidence. Empirical exploration of advancements to such agents through improved retrieval quality, enhanced contextual reasoning, user and role personalization, knowledge governance based on input accuracy, and operational impact with business-security and regulation alignment remains a priority.

References

1. Liu, J., Liu, J., Yang, Y., Wang, J., Wu, W., Zhao, D., & Yan, R. (2022). GNN-encoder: Learning a dual-encoder architecture via graph neural networks for dense passage retrieval. Findings of the Association for Computational Linguistics: EMNLP 2022, 564–575.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

2. Wu, T., Bai, X., Guo, W., Liu, W., Li, S., & Yang, Y. (2023). Modeling fine-grained information via knowledge-aware hierarchical graph for zero-shot entity retrieval. Proceedings of the ACM International Conference on Web Search and Data Mining, 1021–1029.
3. Nagubandi, A. R. (2025). Advanced Predictive Autonomous Agents for Multiportfolio Risk Analytics and Real-Time Enterprise P&L Decisioning: Self-Learning AI Systems for Multi-counterparty Derivatives, Collateral Valuation, and Accounting Reconciliation. Collateral Valuation, and Accounting Reconciliation (December 01, 2025).
4. Wang, D., Liu, S., Wang, H., Grau, B. C., Song, L., Tang, J., & Liu, Q. (2023). An empirical study of retrieval-enhanced graph neural networks. Frontiers in Artificial Intelligence and Applications, 372, 2443–2450.
5. Mavromatis, C., & Karypis, G. (2024). GNN-RAG: Graph neural retrieval for large language model reasoning. arXiv preprint.
6. Mangalampalli, B. M., Bandi, V. D. V. K., Kolla, S. K., & Kumar, M. V. K. (2025). Towards Self-Evolving Healthcare Intelligence: Integrating Advanced Learning Systems with Real-Time Clinical Data Pipelines. Cultura: International Journal of Philosophy of Culture and Axiology, 22(12s), 464-486.
7. Zamiri, M., Qiang, Y., Nikolaev, F., Zhu, D., & Kotov, A. (2024). Benchmark and neural architecture for conversational entity retrieval from a knowledge graph. Proceedings of the ACM Web Conference, 1519–1528.
8. Edge, D., et al. (2024). From local to global: A graph RAG approach to query-focused summarization. arXiv preprint.
9. Mialon, G., et al. (2023). Augmented language models: A survey. Transactions on Machine Learning Research, 1–45.
10. Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Wang, Y., & Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. arXiv preprint.
11. Lewis, P., et al. (2022). Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems, 35, 9459–9474.
12. Mangalampalli, B. M., Bandi, V. D. V. K., Kolla, S. K., & Kumar, M. V. K. (2025). Towards Self-Evolving Healthcare Intelligence: Integrating Advanced Learning Systems with Real-Time Clinical Data Pipelines. Cultura: International Journal of Philosophy of Culture and Axiology, 22(12s), 464-486.
13. Yasunaga, M., et al. (2022). QA-GNN: Reasoning with language models and knowledge graphs for question answering. Proceedings of NAACL-HLT, 535–546.
14. Seenu, A., Aitha, A. R., Gottimukkala, V. R. R., Singireddy, J., Meda, R., & Garapati, R. S. (2025, November). Hybrid Multi-Agent Reinforcement Learning and Blockchain

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

Framework for Real-Time Transaction Integrity in Cloud-Driven Financial Systems. In 2025 IEEE 3rd Global Conference on Wireless Computing and Networking (GCWCN) (pp. 1-6). IEEE.

15. Sun, H., et al. (2022). Think-on-graph: Deep and responsible reasoning of large language models with knowledge graphs. arXiv preprint.
16. Jiang, J., et al. (2023). Graph neural retrieval for multi-hop reasoning. Proceedings of AAAI, 37(4), 4451–4460.
17. Zhu, X., et al. (2023). Knowledge graph enhanced retrieval for enterprise AI systems. Information Processing & Management, 60(4), 103356.
18. Kolla, T. (2025). Generative AI for Intelligent Medical Coding and Healthcare Analytics. International Journal of Advanced Research in Computer Science & Technology (IJARCST), 8(6), 13285-13299.
19. Chen, W., et al. (2024). Graph retrieval augmented generation for enterprise knowledge systems. ACM Transactions on Information Systems, 42(3), 1–29.
20. Li, Y., et al. (2024). Dynamic graph memory networks for intelligent enterprise agents. IEEE Transactions on Neural Networks and Learning Systems, 35(6), 7890–7904.
21. Kurenkov, A., et al. (2023). Modeling dynamic environments with scene graph memory. Advances in Neural Information Processing Systems, 36, 21034–21049.
22. AGENTIC AI FRAMEWORKS FOR AUTONOMOUS RISK DETECTION AND COMPLIANCE REMEDIATION IN ENTERPRISE DATA CENTER OPERATIONS. (2025). Lex Localis - Journal of Local Self-Government, 23(S6), 9672-9697. https://doi.org/10.52152/3f90ak91
23. Pan, S., et al. (2022). Graph learning for intelligent systems: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(11), 7543–7567.
24. Zhang, X., et al. (2022). Knowledge-aware graph retrieval in conversational agents. Proceedings of COLING, 2301–2313.
25. He, K., et al. (2023). Multi-hop graph retrieval with transformer-enhanced graph attention. Proceedings of ACL, 4511–4524.
26. Xu, L., et al. (2023). Graph-based memory augmentation for large language models. arXiv preprint.
27. Yu, S., et al. (2024). Enterprise graph intelligence for autonomous AI agents. IEEE Access, 12, 44821–44839.
28. Kolla, S. K., & Bandi, V. D. V. K. (2025). Autonomous Clinical Monitoring Platforms Using Reinforcement Learning and Deep Neural Networks. International Journal of Advanced Research in Computer Science & Technology (IJARCST), 8(6), 13300-13313.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

29. Yang, H., et al. (2024). Hybrid vector and graph retrieval for enterprise search. Information Sciences, 671, 119870.
30. Mangala, N., & Kolla, S. H. (2025). Real-Time Feature Engineering for Streaming AI Workloads Using PySpark and Azure Event Hubs. International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management, 1(6), 93-105.
31. Cao, R., et al. (2022). Graph reasoning networks for knowledge-intensive retrieval. Proceedings of IJCAI, 3456–3464.
32. Zhou, D., et al. (2023). Retrieval-enhanced graph transformers for intelligent systems. Pattern Recognition, 142, 109641.
33. Guo, J., et al. (2024). Knowledge graph retrieval augmented generation in enterprise workflows. Future Generation Computer Systems, 154, 122–138.
34. Park, J., et al. (2025). Graphiti: Dynamic knowledge graphs for enterprise AI memory. arXiv preprint.
35. Kolla, S. H., & Peddi, R. K. (2024). Designing Governance-Aligned GenAI Pipelines Using Small Language Models for Enterprise Workflow Intelligence. International Journal of Science, Research and Technology, 7(6), 13256-13268.
36. Rasmussen, P., Paliychuk, P., Beauvais, T., Ryan, J., & Chalef, D. (2025). Zep: A temporal knowledge graph architecture for agent memory. arXiv preprint.
37. Li, T., et al. (2025). Neural graph memory for intelligent enterprise automation. Expert Systems with Applications, 245, 122344.
38. Mangala, N., & Kolla, S. H. (2025). Real-Time Feature Engineering for Streaming AI Workloads Using PySpark and Azure Event Hubs. International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management, 1(6), 93-105.
39. Zhao, Q., et al. (2022). Graph neural networks for retrieval and reasoning: A survey. ACM Computing Surveys, 55(7), 1–36.
40. Kumar, V., et al. (2024). Knowledge graph-guided RAG for financial enterprise intelligence. IEEE Access, 12, 67432–67448.
41. Singh, A., et al. (2023). Neural subgraph retrieval for intelligent assistants. Proceedings of EMNLP, 9872–9885.
42. Babaiah, C., Dobriyal, N., Shamila, M., Aitha, A. R., Patel, S. P., & Upodhyay, D. (2025, December). Intelligent Fault Detection and Recovery in Wireless Sensor Networks Using AI. In 2025 IEEE 5th International Conference on ICT in Business Industry & Government (ICTBIG) (pp. 1-6). IEEE.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

43. Patel, M., et al. (2024). Intelligent graph retrieval for enterprise decision systems. Decision Support Systems, 180, 114198.
44. Huang, Z., et al. (2023). Graph-based semantic retrieval for large-scale enterprise agents. Knowledge-Based Systems, 277, 110855.
45. Nagabhyru, K. C., & Kumar, M. V. K. (2025). Generative AI Meets Data Engineering: Automating Code, Query Generation, And Data Insights in Large Scale Enterprises. Query Generation, And Data Insights in Large Scale Enterprises (April 23, 2025).
46. Chen, Y., et al. (2022). Learning graph representations for dense retrieval. Proceedings of SIGIR, 1550–1560.
47. Luo, X., et al. (2024). Graph-augmented retrieval for enterprise AI governance. Information Systems Frontiers, 26(3), 721–739.
48. Wang, Y., et al. (2025). Enterprise graph memory and retrieval for autonomous agents. Journal of Systems Architecture, 152, 103219.
49. Kolla, S. H., & Mattaparthi, R. (2025). Hybrid Gen AI Systems: Integrating Small LMs with Large Language Models for Cost-Efficient Enterprise Automation and Decision Intelligence. International Journal of Research Publications in Engineering, Technology and Management (IJRPETM), 8(6), 13345-13357.
50. Lin, H., et al. (2023). Multi-relational graph retrieval in knowledge-intensive systems. Proceedings of KDD, 2410–2422.
51. Zhao, P., et al. (2022). Neural graph search for question answering. Artificial Intelligence, 311, 103751.
52. Shen, L., et al. (2024). Intelligent graph retrieval pipelines for LLM agents. IEEE Intelligent Systems, 39(4), 33–46.
53. Segireddy, A. R. (2025). AN INTELLIGENT, REAL-TIME DIGITAL FABRIC FOR HEALTHCARE AND FINANCIAL ECOSYSTEMS USING AUTONOMOUS LEARNING AND GENERATIVE SYSTEMS. TPM–Testing, Psychometrics, Methodology in Applied Psychology.
54. Gupta, S., et al. (2025). Adaptive graph retrieval architectures for enterprise-scale agents. ACM Transactions on Knowledge Discovery from Data, 19(1), 1–25.
55. Brown, T., et al. (2022). Knowledge graph retrieval in enterprise automation. Journal of Information Technology, 37(4), 411–427.
56. Feng, J., et al. (2023). Graph memory reasoning for autonomous enterprise agents. Neural Networks, 164, 189–205.
57. Davuluri, P. N. (2020). Improving Data Quality and Lineage in Regulated Financial Data Platforms. Finance and Economics, 1(1), 1-14.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

58. Mehta, R., et al. (2024). Graph-centric retrieval augmentation for regulated enterprise AI. Expert Systems, 41(6), e13301.
59. Das, A., et al. (2022). Learning efficient subgraph retrieval representations. Proceedings of WWW, 1221–1233.
60. Verma, P., et al. (2023). Context-aware graph retrieval for intelligent business systems. IEEE Transactions on Knowledge and Data Engineering, 35(9), 9110–9125.
61. Roy, K., et al. (2024). Enterprise-scale graph retrieval for financial AI ecosystems. Journal of Financial Data Science, 6(2), 88–103.
62. Garapati, R. S. (2025). Real-Time Monitoring and AI-Based Control of Industrial Robots Using Cloud-Hosted Web Applications. Available at SSRN 5612491.
63. Banerjee, D., et al. (2025). Autonomous graph retrieval agents for knowledge-driven enterprises. AI Magazine, 46(1), 56–71.
64. Khandelwal, U., et al. (2022). Retrieval memory for transformer agents. Transactions of the ACL, 10, 973–988.
65. Hu, M., et al. (2024). GraphRAG for intelligent document reasoning. Proceedings of ACL, 6220–6236.
66. Ren, Y., et al. (2025). Neural graph orchestration for multi-agent enterprise intelligence. IEEE Transactions on Services Computing, 18(2), 994–1009.
67. Singh, V., et al. (2023). Graph-enhanced retrieval for enterprise knowledge management. Knowledge and Information Systems, 65(8), 3191–3210.
68. Reddy, V. A. R. (2025). Journal of Rare Cardiovascular Diseases. Health, 5(3), 402-422.
69. Gao, T., et al. (2024). Graph retrieval pipelines for enterprise compliance automation. Information & Management, 61(5), 104011.
70. Zhao, H., et al. (2022). Knowledge-intensive graph search with neural retrievers. Proceedings of AAAI, 36(5), 5412–5420.
71. Kim, J., et al. (2023). Agentic retrieval systems with graph reasoning layers. Proceedings of IJCAI, 4177–4185.
72. Das, N., Qubeb, S. M. P., Amistapuram, K., & Yadav, R. K. (2025). Artificial Inteligence and Data Science. BR Publications.
73. Lee, D., et al. (2024). Intelligent enterprise graph memory for contextual agents. Future Internet, 16(4), 121.
74. Xu, Z., et al. (2025). Self-evolving graph retrieval architectures for enterprise intelligence. IEEE Transactions on Big Data, 11(1), 221–237.
75. Thomas, E., et al. (2023). Knowledge graph retrieval for autonomous service agents. Journal of Web Semantics, 79, 100801.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 03 ISSUE: 04

RECEIVED: OCTOBER 19 REVISED: NOVEMBER 07 ACCEPTED: NOVEMBER 24 PUBLISHED: DECEMBER 22

76. Green, M., et al. (2024). Large language models with graph retrieval memory. Machine Learning with Applications, 18, 100612.
77. Pamisetty, A., Paleti, S., Adusupalli, B., Singireddy, J., Inala, R., & Nagabhyru, K. C. (2025, September). Explainable AI Systems for Credit Scoring and Loan Risk Assessment in Digital Banking Platforms. In 2025 IEEE 13th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS) (pp. 1478-1483). IEEE.
78. Kapoor, N., et al. (2025). Enterprise-grade graph retrieval augmented reasoning systems. Expert Systems with Applications, 252, 123445.
79. Bansal, S., Sistla, S. S., Arikatala, A., & Schreiber, S. (2025). Planning agents on an ego-trip: Leveraging hybrid ego-graph ensembles for improved tool retrieval in enterprise task planning. arXiv preprint.