Section 5: Agent Memory

2026 Spring

LLM Agents Foundation & Applications

Student Team

20260310

Roadmap: three papers

- 1. **Memory in the Age of AI Agents: A Survey**
- 2. Zero-RAG: Towards RAG with Zero Redundant Knowledge
- 3. From Local to Global: A GraphRAG Approach to Query-Focused Summarization

Memory in the Age of AI Agents: A Survey

Authors: Hu et al.

Motivation

The literature on agent memory is fragmented

- Many works study “memory” but with very different meanings
- Terms such as:
  - episodic memory
  - RAG memory
  - KV memory
- Existing surveys do not capture recent developments

This paper proposes a unified taxonomy of agent memory:
Forms – Functions – Dynamics

Motivation

UNIVERSITY of VIRGINIA

Key Questions

❶ How is *agent memory* defined, and how does it relate to related concepts such as LLM memory, retrieval-augmented generation (RAG), and context engineering?

❷ **Forms:** What architectural or representational forms can agent memory take?

❸ **Functions:** Why is agent memory needed, and what roles or purposes does it serve?

❹ **Dynamics:** How does agent memory operate, adapt, and evolve over time?

❺ What are the promising frontiers for advancing agent memory research?

Agent & Memory Formalization

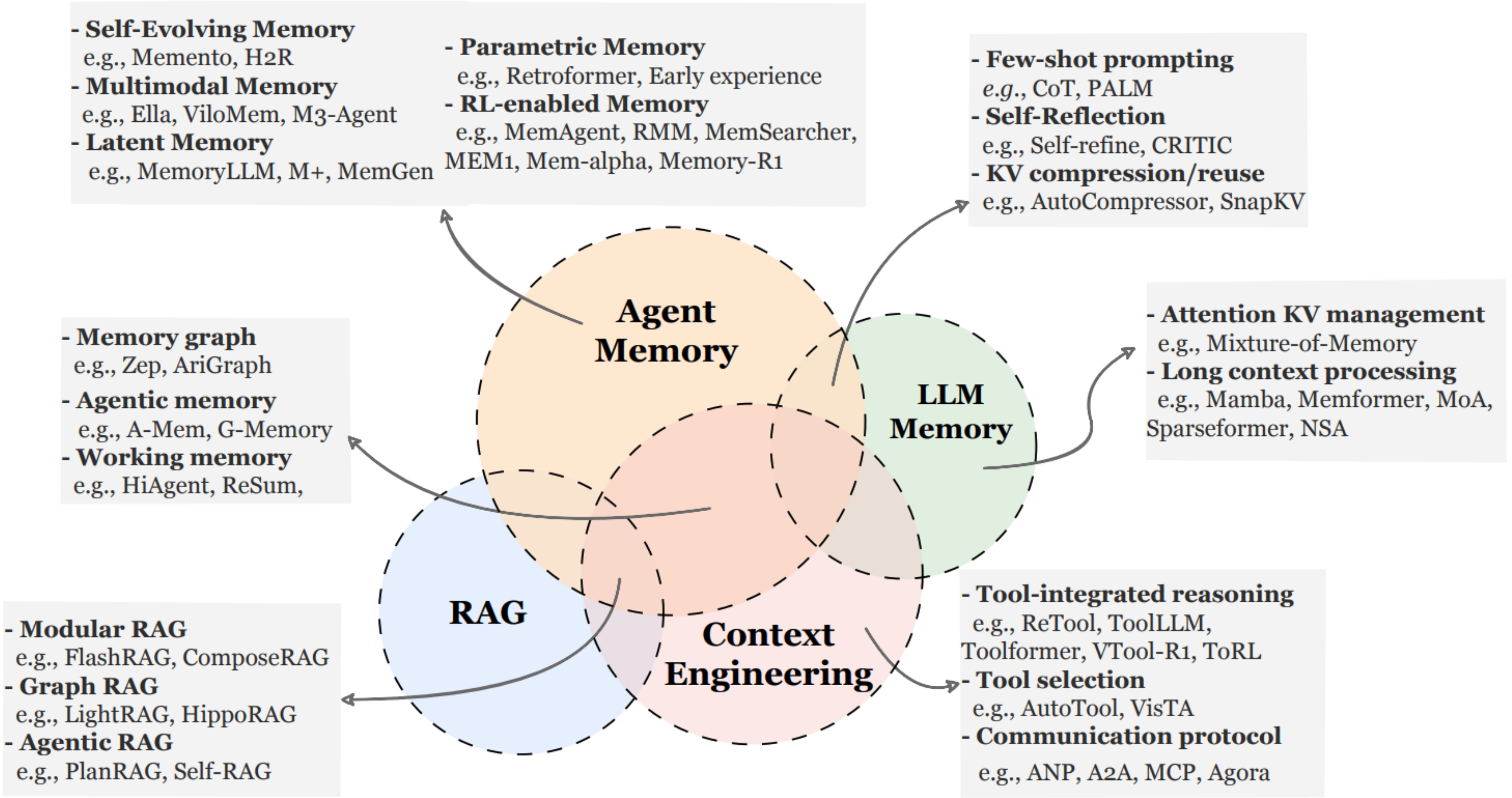


Figure 2 Conceptual comparison of **Agent Memory** with **LLM Memory**, **RAG**, and **Context Engineering**. The diagram illustrates shared technical implementations (e.g., KV reuse, graph retrieval) while highlighting fundamental distinctions: unlike the architectural optimizations of LLM Memory, the static knowledge access of RAG, or the transient resource management of Context Engineering, Agent Memory is uniquely characterized by its focus on maintaining a persistent and self-evolving cognitive state that integrates factual knowledge and experience. The listed categories and examples are illustrative rather than strictly parallel, serving as representative reference points to clarify conceptual relationships rather than to define a rigid taxonomy.

Forms of Memory

UNIVERSITY of VIRGINIA

Three Major Memory Forms

1. **Token-level Memory** (Section 3.1): Memory organized as explicit and discrete units that can be individually accessed, modified, and reconstructed. These units remain externally visible and can be stored in a structured form over time.
2. **Parametric Memory** (Section 3.2): Memory stored within the model parameters, where information is encoded through the statistical patterns of the parameter space and accessed implicitly during forward computation.
3. **Latent Memory** (Section 3.3): Memory represented in the model's internal hidden states, continuous representations, or evolving latent structures. It can persist and update during inference or across interaction cycles, capturing context-dependent internal states.

Forms of Memory

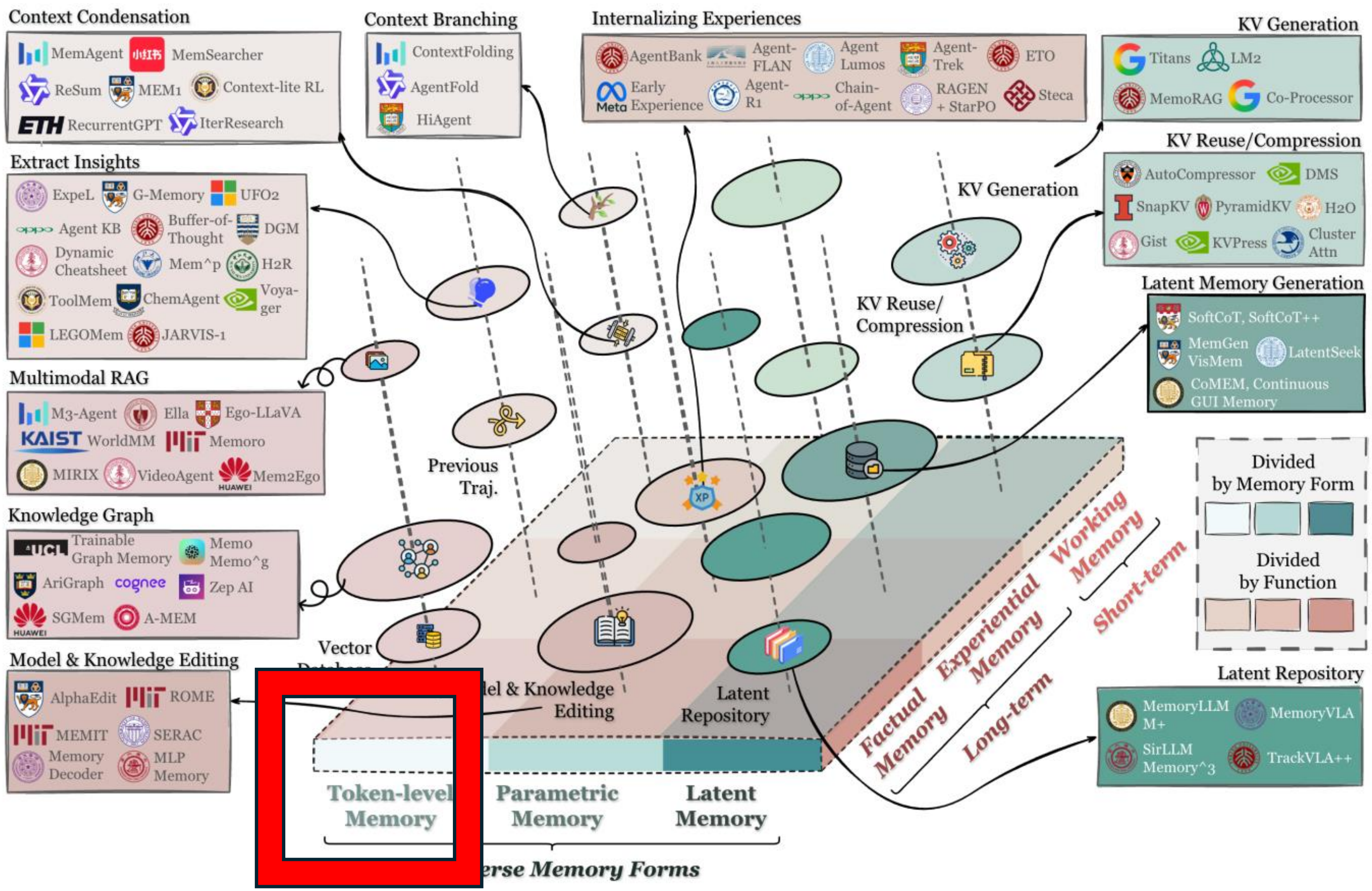


Figure 1 Overview of agent memory organized by the unified taxonomy of *forms* (Section 3), *functions* (Section 4), and *dynamics* (Section 5). The diagram positions memory artifacts by their dominant form and primary function. It further maps representative systems into this taxonomy to provide a consolidated landscape.

Forms of Memory – Token-level

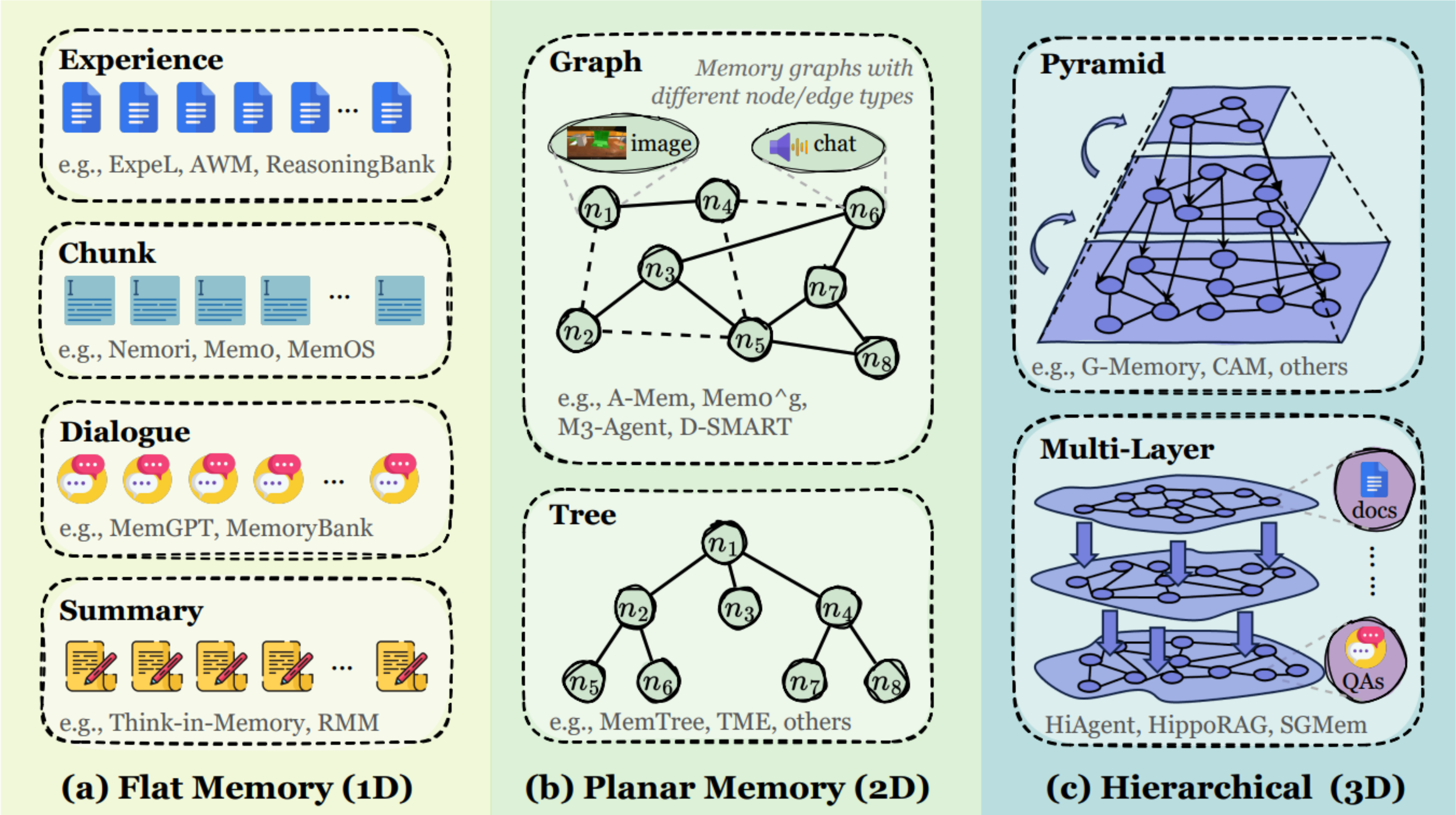


Figure 3 Taxonomy of token-level memory organized by topological complexity and dimensionality: (a) **Flat Memory (1D)** stores information as linear sequences or independent clusters without explicit inter-unit topology, commonly used for *Chunk* sets, *Dialogue* logs, and *Experience* pools. (b) **Planar Memory (2D)** introduces a single-layer structured layout where units are linked via **Tree** or **Graph** structures to capture relational dependencies, supporting diverse node types such as images and chat records. (c) **Hierarchical Memory (3D)** employs multi-level forms, such as **Pyramids** or **Multi-layer** graphs, to facilitate vertical abstraction and cross-layer reasoning between different data granularities, such as raw docs and synthesized QAs.

Forms of Memory – Token-level

UNIVERSITY of VIRGINIA

Definition of Token-level Memory

Token-level memory stores information as persistent, discrete units that are externally accessible and inspectable. The token here is a broad representational notion: beyond text tokens, it includes visual tokens, audio frames—any discrete element that can be written, retrieved, reorganized, and revised outside model parameters.

Three Major Types of Token-level Memory

1. **Flat Memory (1D)**: No explicit inter-unit topology. Memories are accumulated as sequences or bags of units (e.g., snippets, trajectories, chunks)
2. **Planar Memory (2D)**: A structured but single-layer organization within one plane: units are related by a graph, tree, table and so on, with no cross-layer relations. The structure is explicit, but not layered.
3. **Hierarchical Memory (3D)**: Structured across multiple layers with inter-layer links, forming a volumetric or stratified memory

Table 1 Comparison of representative token-level memory methods. We categorize existing works into three groups based on their topological complexity: **Flat Memory (1D)** for linear or independent records, **Planar Memory (2D)** for structured single-layer graphs/trees, and **Hierarchical Memory (3D)** for multi-level architectures. Methods are characterized across four dimensions: (1) **Multi** indicates multimodal capability, where ✔ denotes support for modalities beyond text (e.g., visual) and ✗ implies text-only; (2) **Type** identifies the specific functional category of the memory (e.g., *Fact* for factual memory, *Exp* for experiential memory, *Work* for working memory); (3) **Memory Form** details the content of the stored units; and (4) **Task** lists the primary application domains.

Method	Multi	Type	Memory Form	Task
			Flat Memory Models	
Reflexion (Shinn et al., 2023b)	✗	E&W	Trajectory as short-term and feedback as long-term	QA, Reasoning, Coding
Memento (Zhou et al., 2025a)	✗	Exp	Trajectory case (success/failure).	Reasoning
JARVIS-1 (Wang et al., 2025q)	✔	Exp	Plan-environment pairs.	Game
Expel (Zhao et al., 2024)	✗	Exp	Insights and few-shot examples.	Reasoning
Buffer of Thoughts (Yang et al., 2024b)	✗	Exp	High-level thought-templates.	Game, Reasoning, Coding
SAGE (Liang et al., 2025)	✗	Exp	Dual-store with forgetting mechanism.	Game, Reasoning, Coding
ChemAgent (Tang et al., 2025c)	✗	Exp	Structured sub-tasks and principles.	Chemistry
AgentKB (Tang et al., 2025d)	✗	Exp	5-tuple experience nodes.	Coding, Reasoning
H^2R (Ye et al., 2025b)	✗	Exp	Planning and Execution layers.	Game, Embodied Simulation
AWM (Wang et al., 2024m)	✗	Exp	Abstracted universal workflows.	Web
PRINCIPLES (Kim et al., 2025a)	✗	Exp	Rule templates from self-play.	Emotional Companion
ReasoningBank (Ouyang et al., 2025)	✗	Exp	Transferable reasoning strategy items.	Web
Voyager (Wang et al., 2024b)	✔	Exp	Executable skill code library.	Game
DGM (Zhang et al., 2025i)	✗	Exp	Recursive self-modifiable codebase.	Coding
Memp (Fang et al., 2025d)	✗	Exp	Instructions and abstract scripts.	Embodied Simulation, Travel Planning

Planar Memory Models

D-SMART (Lei et al., 2025)	✗	Fact	Structured memory with reasoning trees.	Long-conv QA
Reflexion (Shinn et al., 2023b)	✗	Work	Reflective text buffer from experiences.	QA, Reasoning, Coding
PREMem (Kim et al., 2025b)	✗	Fact	Dynamic cross-session linked triples.	Long-conv QA
Query Reconstruct (Xu et al., 2025b)	✗	Exp	Logic graphs built from knowledge bases.	KnowledgeGraph QA
KGT (Sun et al., 2024)	✗	Fact	KG node from query and feedback.	QA
Optimus-1 (Li et al., 2024d)	✔	F&E	Knowledge graph and experience pool.	Game
SALI (Pan et al., 2024)	✔	Exp	Topological graph with spatial nodes	Navigation
HAT (A et al., 2024)	✗	Fact	Hierarchical aggregate tree.	Long-conv QA
MemTree (Rezazadeh et al., 2025c)	✗	Fact	Dynamic hierarchical conversation tree.	Long-conv QA
TeaFarm (iunn Ong et al., 2025)	✗	Fact	Causal edges connecting memories.	Long-conv QA
COMET (Kim et al., 2024b)	✗	Fact	Context-aware memory through graph.	Long-conv QA
Intrinsic Memory (Yuen et al., 2025)	✗	Fact	Private internal and shared external mem.	Planning
A-MEM (Xu et al., 2025c)	✗	Fact	Card-based connected mem.	Long-conv QA
Ret-LLM (Modarressi et al., 2023)	✗	Fact	Triplet table and LSH vectors.	QA
HuaTuo (Wang et al., 2023a)	✗	Fact	Medical Knowledge Graph.	Medical QA
M3-Agent (Long et al., 2025)	✔	Fact	Multimodal nodes in graph structure.	Embodied QA
EMem (Zhou and Han, 2025a)	✗	Fact	Event-centric alternative with pagerank.	Long-conv QA
WorldMM (Yeo et al., 2025)	✔	Fact	Multiple complementary memories.	Video Understanding
Memoria (Sarin et al., 2025)	✗	Fact	Knowledge-graph profile and summary.	Long-conv QA
LingoEDU (Zhou et al., 2026)	✗	Fact	Relation tree of Elementary Discourse Units.	Long-conv QA

Hierarchical Memory Models

GraphRAG (Edge et al., 2025)	✗	Fact	Multi-level community graph indices.	QA, Summarization
H-Mem (Sun and Zeng, 2025)	✗	Fact	Decoupled index layers and content layers.	Long-conv QA
EMG-RAG (Wang et al., 2024l)	✗	Fact	Three-tiered memory graph.	QA
G-Memory (Zhang et al., 2025c)	✗	Exp	Query-centric three-layer graph structure.	QA, Game, Embodied Task
Zep (Rasmussen et al., 2025)	✗	Fact	Temporal Knowledge Graphs.	Long-conv QA
SGMem (Wu et al., 2025h)	✗	Fact	Chunk Graph and Sentence Graph.	Long-conv QA
HippoRAG (Gutierrez et al., 2024)	✗	Fact	Knowledge with query nodes.	QA
HippoRAG 2 (Gutiérrez et al., 2025)	✗	Fact	KG with phrase and passage.	QA
AriGraph (Anokhin et al., 2024)	✗	Fact	Semantic and Episodic memory graph.	Game
Lyfe Agents (Kaiya et al., 2023)	✗	Fact	Working, Short & Long-term layers.	Social Simulation
CAM (Li et al., 2025g)	✗	Fact	Multilayer graph with topic.	Doc QA
HiAgent (Hu et al., 2025a)	✗	E&W	Goal graphs with recursive cluster.	Agentic Tasks
ILM-TR (Tang et al., 2024)	✗	Fact	Hierarchical Memory tree.	Long-context
CompassMem (Hu et al., 2026b)	✗	Fact	Hierarchical event-centric Memory.	QA
MAGMA (Jiang et al., 2026)	✗	Fact	Semantic, temporal, causal, entity graphs.	Long-conv QA
EverMemOS (Hu et al., 2026a)	✗	Fact	Reusable memories covering multi types.	Long-conv QA
RGMem (Tian et al., 2025a)	✗	Fact	Renormalization Group-based memory.	Long-conv QA
MemVerse (Liu et al., 2025e)	✓	Fact	Multimodal hierarchical knowledge graphs.	Reasoning, QA

Forms of Memory

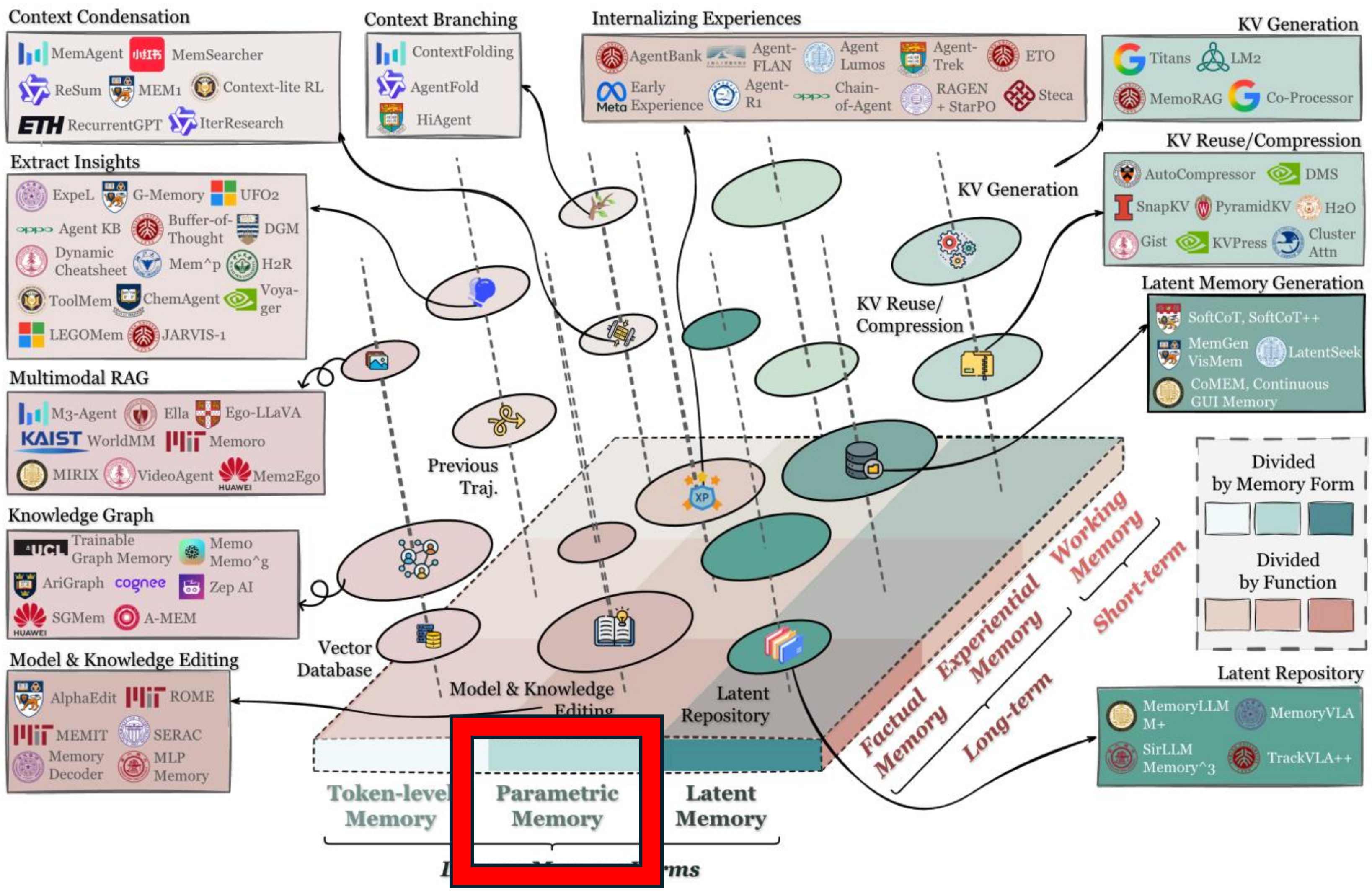


Figure 1 Overview of agent memory organized by the unified taxonomy of *forms* (Section 3), *functions* (Section 4), and *dynamics* (Section 5). The diagram positions memory artifacts by their dominant form and primary function. It further maps representative systems into this taxonomy to provide a consolidated landscape.

Forms of Memory -- Parametric

Two Major Types of Parametric Memory

1. **Internal Parametric Memory**: Memory encoded within the original parameters of the model (e.g., weights, biases). These methods directly adjust the base model to incorporate new knowledge or behavior.
2. **External Parametric Memory**: Memory stored in additional or auxiliary parameter sets, such as adapters, LoRA modules, or lightweight proxy models. These methods introduce new parameters to carry memory without modifying the original model weights.

Method	Type	Task	Optimization
		I. Internal Parametric Memory	
(a) Pre-Train Phase			
TNL (Qin et al., 2024b)	Working	QA, Reasoning	SFT
StreamingLLM (Xiao et al., 2024)	Working	QA, Reasoning	SFT
LMLM (Zhao et al., 2025b)	Factual	QA, Factual Gen	SFT
HierMemLM (Pouransari et al., 2025)	Factual	QA, Language Modeling	SFT
Function Token (Zhang et al., 2025o)	Factual	Language Modeling	Pretrain
(b) Mid-Train Phase			
Agent-Founder (Su et al., 2025)	Experiential	Tool Calling, Deep Research	SFT
Early Experience (Zhang et al., 2025k)	Experiential	Tool Calling, Embodied Simulation, Reasoning, Web	SFT
(c) Post-Train Phase			
Character-LM (Shao et al., 2023)	Factual	Role Playing	SFT
CharacterGLM (Zhou et al., 2024a)	Factual	Role Playing	SFT
SELF-PARAM (Wang et al., 2025o)	Factual	QA, Recommendation	KL Tuning
Room (Kim et al., 2023b)	Experiential	Embodied Task	RL
KnowledgeEditor (Cao et al., 2021)	Factual	QA, Fact Checking	FT
Mend (Mitchell et al., 2022)	Factual	QA, Fact Checking, Model Editing	FT
PersonalityEdit Mao et al. (2024)	Factual	QA, Model Editing	FT, PE
APP (Ma et al., 2024)	Factual	QA	FT
DINM (Wang et al., 2024c)	Experiential	QA, Detoxification	FT
AlphaEdit (Fang et al., 2025c)	Factual	QA	FT
		II. External Parametric Memory	
(a) Adapter-based Modules			
MLP-Memory (Wei et al., 2025d)	Factual	QA, Classification, Textual Entailment	SFT
K-Adapter (Wang et al., 2021)	Factual	QA, Entity Typing, Classification	SFT
WISE (Wang et al., 2024e)	Factual	QA, Hallucination Detection	SFT
ELDER (Li et al., 2025d)	Factual	Model Editing	SFT
T-Patcher (Huang et al., 2023)	Factual	QA	FT
Sparse Memory FT (Lin et al., 2025a)	Factual	QA	SFT
Memory Decoder (Cao et al., 2025a)	Factual	QA, Language Modeling	SFT
MemLoRA (Bini et al., 2025)	Factual	QA	SFT
(b) Auxiliary LM-based Modules			
MAC (Tack et al., 2024)	Factual	QA	SFT
Retroformer (Yao et al., 2024a)	Experiential	QA, Web Navigation	RL

UNIVERSITY of VIRGINIA

Table 2 Taxonomy of parametric memory methods. We categorize existing works based on the storage location relative to the core model: **Internal Parametric Memory** embeds knowledge directly into the original weights, while **External Parametric Memory** isolates information within auxiliary parameter sets. Based on the training **phase**, we performed a secondary classification of the articles. Methods are compared across three technical dimensions: (1) **Type** defines the nature of the memory, (2) **Task** specifies the target downstream application, and (3) **Optimization** denotes the optimization strategy, such as SFT, FT (fine-tuning) , and PE (prompt engineering).

Forms of Memory

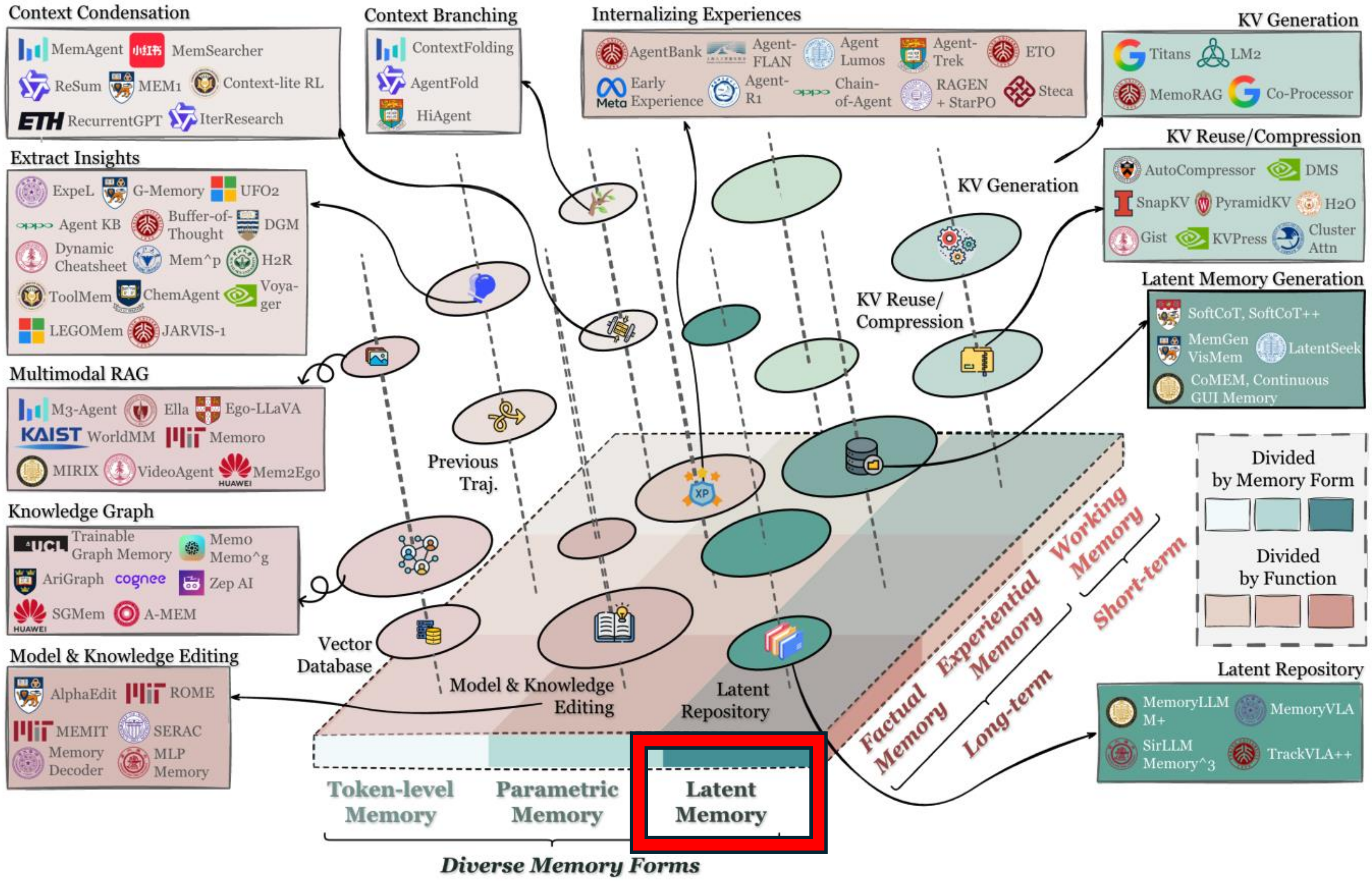


Figure 1 Overview of agent memory organized by the unified taxonomy of *forms* (Section 3), *functions* (Section 4), and *dynamics* (Section 5). The diagram positions memory artifacts by their dominant form and primary function. It further maps representative systems into this taxonomy to provide a consolidated landscape.

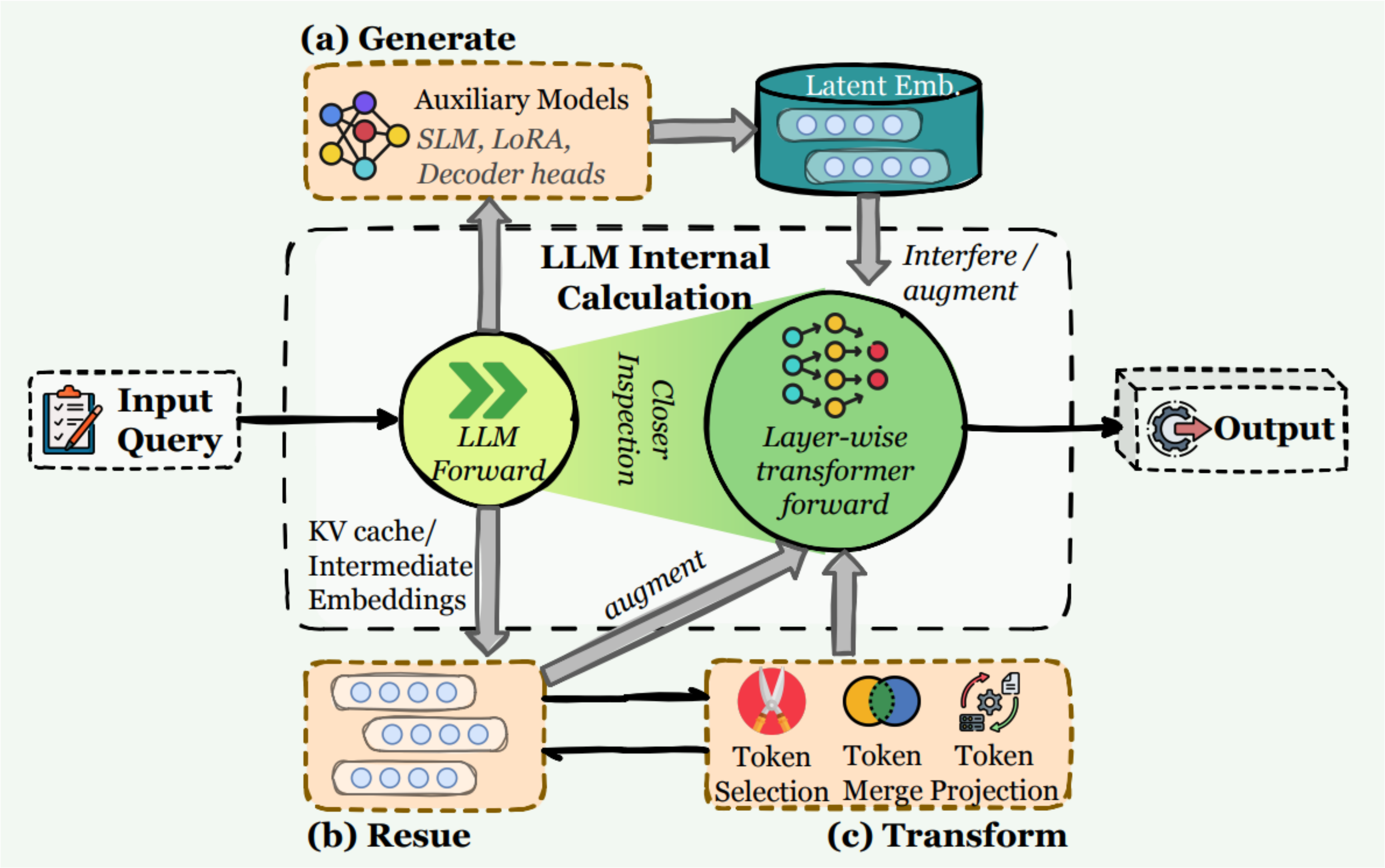


Figure 4 Overview of Latent Memory integration in LLM agents. Unlike explicit text storage, latent memory operates within the model's internal representational space. The framework is categorized by the origin of the latent state: (a) **Generate**, where auxiliary models synthesize embeddings to interfere with or augment the LLM's forward pass; (b) **Reuse**, which directly propagates prior computational states such as KV caches or intermediate embeddings; and (c) **Transform**, which compresses internal states through token selection, merging, or projection to maintain efficient context.

Forms of Memory -- Latent

Definition of Latent Memory

Latent memory refers to memory that is carried implicitly in the model's internal representations (e.g., KV cache, activations, hidden states, latent embeddings), rather than being stored as explicit, human-readable tokens or dedicated parameter sets.

Three Major Types of Latent Memory

1. **Generate**: latent memory is produced by an independent model or a module, and then supplied to the agent as reusable internal representations.
2. **Reuse**: latent memory is directly carried over from prior computation, most prominently KV-cache reuse (within or across turns), as well as recurrent or stateful controllers that propagate hidden states.
3. **Transform**: existing latent state is transformed into new representations (e.g., distillation, pooling, or compression), so the agent can retain essentials while reducing latency and context footprint.

Forms of Memory -- Latent

University of Virginia

Method	Form	Type	Task
	I. Generate		
(a) Single Modal			
Gist (Mu et al., 2023)	Gist Tokens	Working	Long-context Compression
Taking a Deep Breath (Luo et al., 2024)	Sentinel Tokens	Working	Long-context QA
SoftCoT (Xu et al., 2025d)	Soft Tokens	Working	Reasoning
CARE (Choi et al., 2025)	Memory Tokens	Working	QA, Fact Checking
AutoCompressor (Chevalier et al., 2023)	Summary Vectors	Working	QA, Compression
MemoRAG (Qian et al., 2025)	Global Semantic States	Working	QA, Summary
MemoryLLM (Wang et al., 2024j)	Persistent Tokens	Factual	Long-conv QA, Model Editing
M+ (Wang et al., 2025n)	Cross-layer Token Pools	Factual	QA
LM2 (Kang et al., 2025b)	Matrix Slots	Working	QA, Reasoning
Titans (Behrouz et al., 2025b)	Neural Weights (MLP)	Working	QA, Language Modeling
MemGen (Zhang et al., 2025d)	LoRA Fragments	Working, Exp.	QA, Math, Code, Embodied Task, Reasoning
EMU (Na et al., 2024)	Embeddings w/ Returns	Factual	Game
TokMem (Wu et al., 2025j)	Memory Tokens	Exp.	Funcation calling
Nested Learning (Behrouz et al., 2025a)	Nested Optimization	Factual	Language Modeling
Memoria (Park and Bak, 2024)	Three memory layers with engrams	Factual	Language Modeling
(b) Multi-Modal			
CoMem (Wu et al., 2025d)	Multimodal Embeddings	Factual	Multimodal QA
ACM (Wu et al., 2025e)	Trajectory Embeddings	Working	Web
Time-VLM (Zhong et al., 2025)	Patch Embeddings	Working	Video Understanding
Mem Augmented RL (Mezghani et al., 2022)	Novelty State Encoder	Working	Visual Navigation
MemoryVLA (Shi et al., 2025a)	Perceptual States	Factual, Working	Embodied Task
XMem (Cheng and Schwing, 2022)	Key-Value Embeddings	Working	Video Segmentation
	II. Reuse		
Memorizing Transformers (Wu et al., 2022)	External KV Cache	Working	Language Modeling
SirLLM (Yao et al., 2024b)	Entropy-selected KV	Factual	Long-conv QA
Memory^3 (Yang et al., 2024a)	Critical KV Pairs	Factual	QA
FOT (Tworkowski et al., 2023)	Memory-Attention KV	Working	QA, Few-shot learning, Language Modeling
LONGMEM (Wang et al., 2023b)	Residual SideNet KV	Working	Language Modeling and Understanding
	III. Transform		
Scissorhands (Liu et al., 2023b)	Pruned KV	Working	Image classification & generation
SnapKV (Li et al., 2024b)	Aggregated Prefix KV	Working	Language Modeling
PyramidKV (Cai et al., 2024)	Layer-wise Budget	Working	Language Modeling
RazorAttention (Tang et al., 2025a)	Compensated Window	Working	Language Modeling
H2O (Zhang et al., 2023)	Heavy Hitter Tokens	Working	QA, Language Modeling
R^{3}Mem (Wang et al., 2025k)	Virtual memory tokens with reversible compression	Working	QA, Language Modeling

Table 3 Taxonomy of latent memory methods. We categorize existing works based on the origin of the latent state: **Generate** synthesizes memory via auxiliary modules, **Reuse** propagates internal computational states, and **Transform** compresses, modifies or restructs existing latent state. Methods are compared across three technical dimensions: (1) **Form** specifies the specific data type of the latent memory, (2) **Type** defines the nature of the recorded content (e.g., Working, Factual, and Experiential), and (3) **Task** denotes the target downstream application.

Forms of Memory

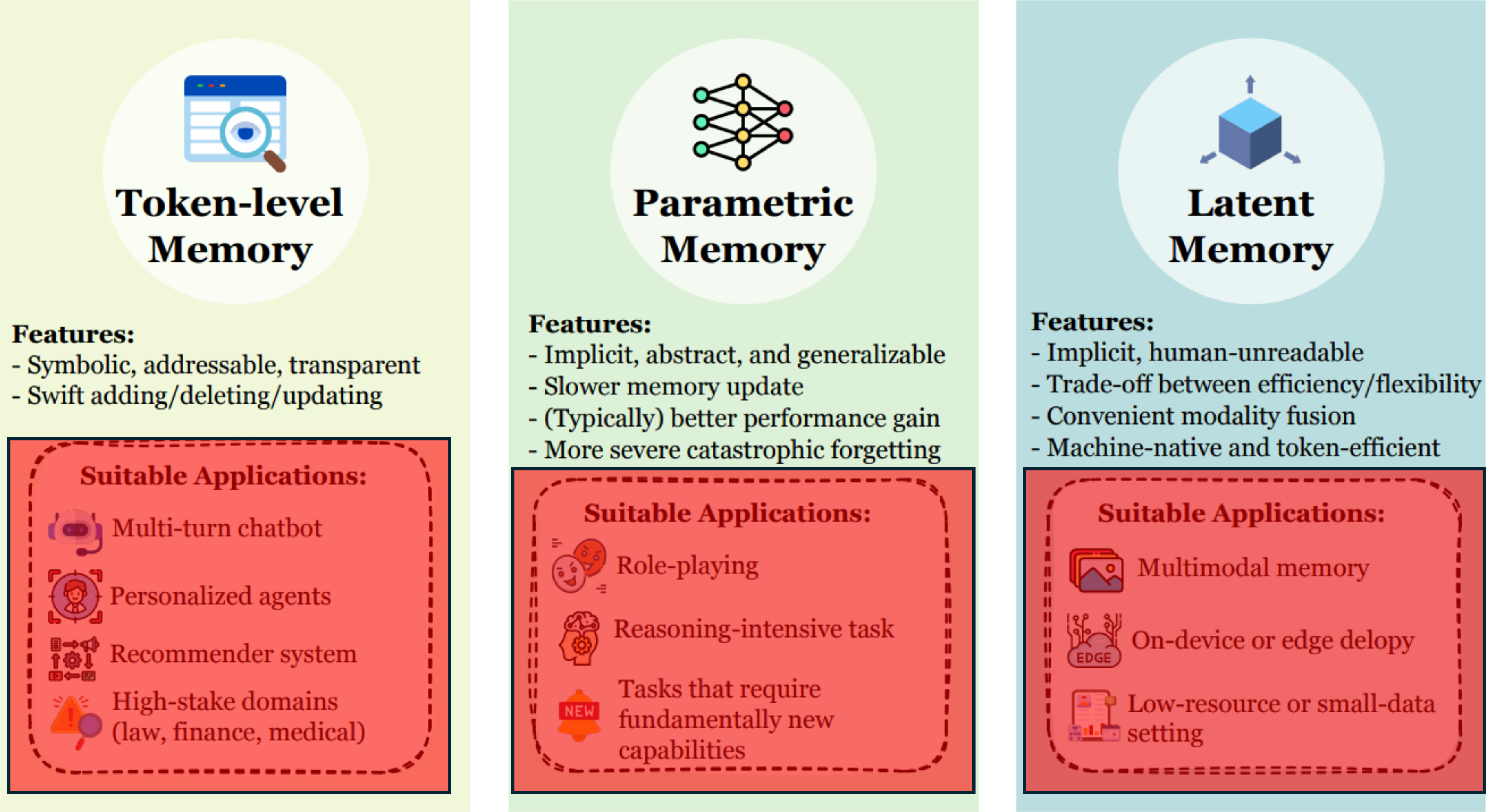


Figure 5 Overview of three complementary memory paradigms for LLM agents. Token-level, parametric, and latent memories differ in their representational form, update dynamics, interpretability, and efficiency, leading to distinct strengths, limitations, and application domains in long-horizon and interactive agent systems.

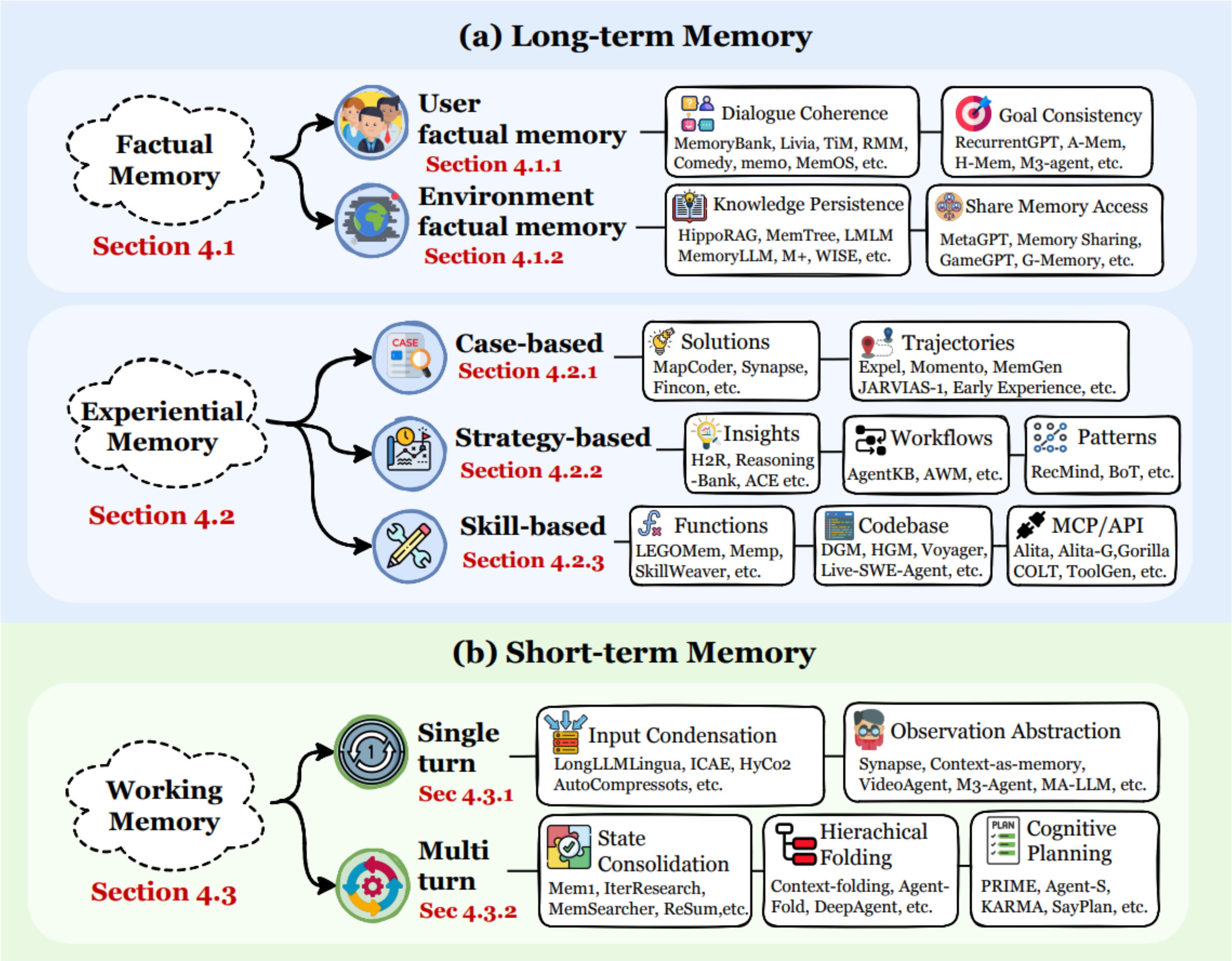


Figure 6 The functional taxonomy of agent memory. We organize memory capabilities based on their *functions* (purpose) into three primary pillars spanning two temporal domains: (1) **Factual Memory** serves as a persistent declarative knowledge base to ensure interaction *consistency*, *coherence*, and *adaptability*; (2) **Experiential Memory** encapsulates procedural knowledge to enable *continual learning* and *self-evolution* across episodes; and (3) **Working Memory** provides mechanisms for the active management of transient context.

Functions of Memory

Three Primary Memory Functions

1. **Factual Memory** (Section 4.1): The agent's declarative knowledge base, established to ensure consistency, coherence, and adaptability by recalling explicit facts, user preferences, and environmental states. This system answers the question: "What does the agent know?"
2. **Experiential Memory** (Section 4.2): The agent's procedural and strategic knowledge, accumulated to enable continual learning and self-evolution by abstracting from past trajectories, failures, and successes. This system answers: "How does the agent improve?"
3. **Working Memory** (Section 4.3): The agent's capacity-limited, dynamically controlled scratchpad for active context management during a single task or session. This system answers: "What is the agent thinking about now?"

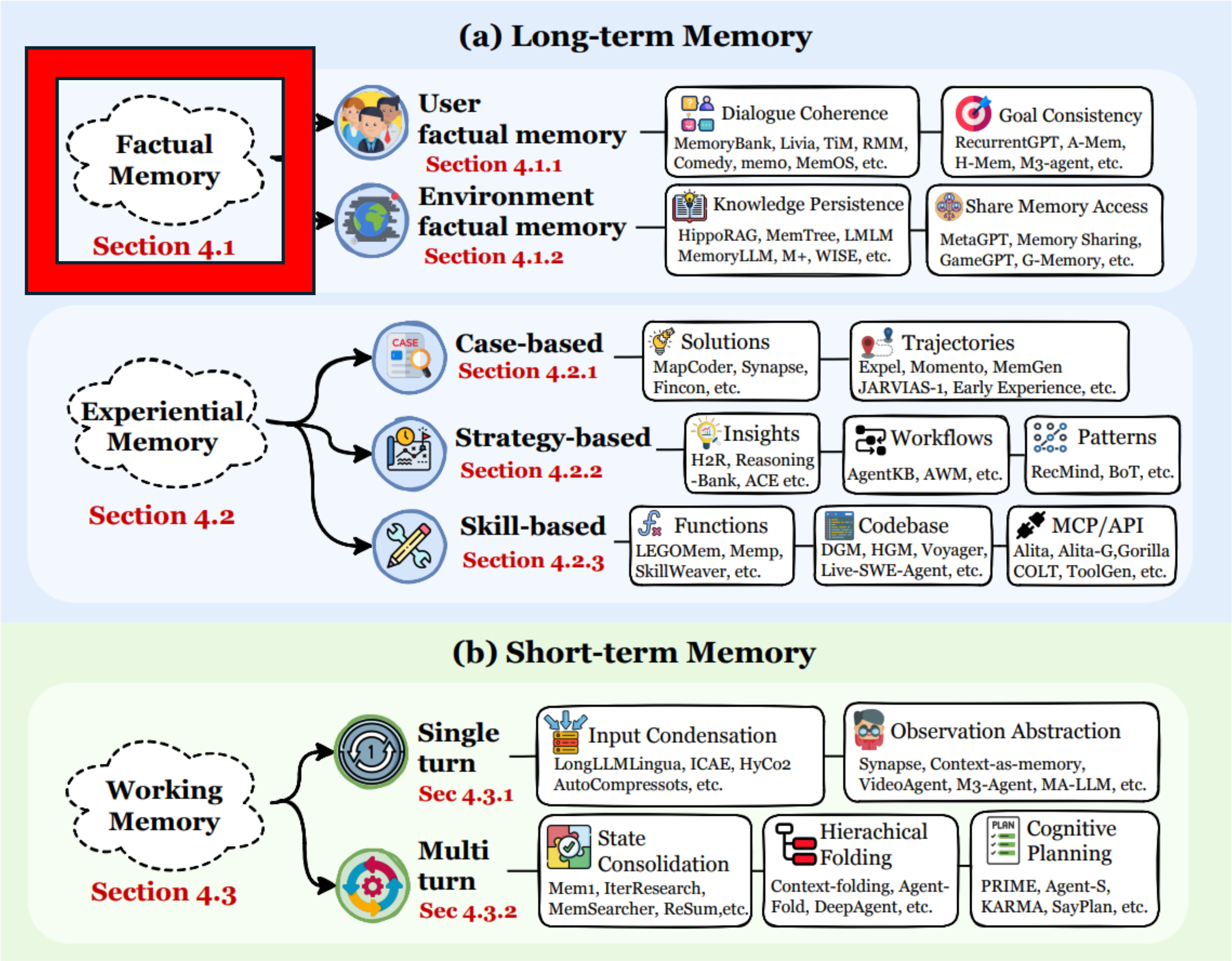


Figure 6 The functional taxonomy of agent memory. We organize memory capabilities based on their *functions* (purpose) into three primary pillars spanning two temporal domains: (1) **Factual Memory** serves as a persistent declarative knowledge base to ensure interaction *consistency*, *coherence*, and *adaptability*; (2) **Experiential Memory** encapsulates procedural knowledge to enable *continual learning* and *self-evolution* across episodes; and (3) **Working Memory** provides mechanisms for the active management of transient context.

Functions of Memory – Factual

Two Types of Factual Memory

- **User factual memory** (Section 4.1.1) denotes facts that sustain the consistency of interactions between humans and agents, including identities, stable preferences, task constraints, and historical commitments.
- **Environment factual memory** (Section 4.1.2) denotes facts that sustain consistency with respect to the external world, such as document states, resource availability, and the capabilities of other agents.

Functions of Memory – Factual

Table 4 Taxonomy of factual memory methods. We categorize existing works based on the primary target entity: **User Factual Memory** focuses on sustaining interaction consistency, while **Environment Factual Memory** ensures consistency with the external world. Methods are compared across three technical dimensions: (1) **Carrier** (Section 3) identifies the storage medium, (2) **Structure** follows the taxonomy of token-level memory (Section 3.1), and (3) **Optimization** denotes the integration strategy, where *PE* encompasses prompt engineering and inference-time techniques without parameter updates, distinct from gradient-based methods like *SFT* and *RL*.

Method	Carrier	Structure	Task	Optimization
		I. User factual Memory		
(a) Dialogue Coherence				
MemGPT (Packer et al., 2023b)	Token-level	1D	Long-term dialogue	PE
TiM (Liu et al., 2023a)	Token-level	2D	QA	PE
MemoryBank (Zhong et al., 2024)	Token-level	1D	Emotional Companion	PE
AI Persona (Wang et al., 2024f)	Token-level	1D	Emotional Companion	PE
Encode-Store-Retrieve (Shen et al., 2024)	Token-level	1D	Multimodal QA	PE

Continued on next page

Method	Carrier	Form	Task	Optimization
Livia (Xi and Wang, 2025)	Token-level	1D	Emotional Companion	PE
mem0 (Chhikara et al., 2025)	Token-level	1D	Long-term dialogue, QA	PE
RMM (Tan et al., 2025c)	Token-level	2D	Personalization	PE, RL
D-SMART (Lei et al., 2025)	Token-level	2D	Reasoning	PE
Comedy (Chen et al., 2025d)	Token-level	1D	Summary, Compression, QA	PE
MEMENTO (Kwon et al., 2025)	Token-level	1D	Embodied, Personalization	PE
O-Mem (Wang et al., 2025g)	Token-level	3D	Personalized Dialogue	PE
DAM-LLM (Lu and Li, 2025)	Token-level	1D	Emotional Companion	PE
MemInsight (Salama et al., 2025)	Token-level	1D	Personalized Dialogue	PE
EMem (Zhou and Han, 2025a)	Token-level	1D	Personalized Dialogue	PE
RGMem (Tian et al., 2025a)	Token-level	1D	Long-conv QA	PE
Memoria (Sarin et al., 2025)	Token-level	1D	Long-conv QA	PE
MemVerse (Liu et al., 2025e)	✔	Fact	Multimodal hierarchical knowledge graphs.	Reasoning, QA
(b) Goal Consistency				
RecurrentGPT (Zhou et al., 2023b)	Token-level	1D	Long-Context Generation, Personalized Interactive Fiction	PE
Memolet (Yen and Zhao, 2024)	Token-level	2D	QA, Document Reasoning	PE
MemGuide (Du et al., 2025b)	Token-level	1D	Long-conv QA	PE, SFT
SGMem (Wu et al., 2025h)	Token-level	2D	Long-context	PE
A-Mem (Xu et al., 2025c)	Token-level	2D	QA, Reasoning	PE
M3-agent (Long et al., 2025)	Token-level	2D	Multimodal QA	PE, SFT
WorldMM (Yeo et al., 2025)	Token-level	1D	Multimodal QA	PE
EverMemOS (Hu et al., 2026a)	Token-level	1D	Long-conv QA	PE
II. Environment factual Memory				
(a) Knowledge Persistence				
MemGPT (Packer et al., 2023b)	Token-level	1D	Document QA	PE
CALYPSO (Zhu et al., 2023)	Token-level	1D	Tabletop Gaming	PE
AriGraph (Anokhin et al., 2024)	Token-level	3D	Game, Multi-op QA	PE
HippoRAG (Gutierrez et al., 2024)	Token-level	3D	QA	PE
WISE (Wang et al., 2024e)	Parametric	/	Document Reasoning, QA	SFT
MemoryLLM (Wang et al., 2024j)	Parametric	/	Document Reasoning	SFT
Memoria (Park and Bak, 2024)	latent	/	Language Modeling	PE
Zep (Rasmussen et al., 2025)	Token-level	3D	Document analysis	PE
MemTree (Rezazadeh et al., 2025c)	Token-level	2D	Document Reasoning, Dialogue	PE
LMLM (Zhao et al., 2025b)	Token-level	1D	QA	SFT
M+ (Wang et al., 2025n)	Latent	/	Document Reasoning, QA	SFT
CAM (Li et al., 2025g)	Token-level	3D	Multi-hop QA	SFT, RFT
MemAct (Zhang et al., 2025r)	Token-level	1D	Multi-obj QA	RL
Mem-α (Wang et al., 2025p)	Token-Level	1D	Document Reasoning	RL
WebWeaver (Li et al., 2025m)	Token-level	1D	Deep Research	SFT
MemLoRA (Bini et al., 2025)	Parametric	/	QA	SFT
Memory Decoder (Cao et al., 2025a)	Parametric	/	QA, Language Modeling	SFT
(b) Shared Access				
GameGPT (Chen et al., 2023b)	Token-level	1D	Game Development	PE
Generative Agent (Park et al., 2023)	Token-level	2D	Social Simulation	PE
S³ (Gao et al., 2023a)	Token-level	1D	Social Simulation	PE
Memory Sharing (Gao and Zhang, 2024a)	Token-level	1D	Document Reasoning	PE
MetaGPT (Hong et al., 2024)	Token-level	1D	Software Development	PE
G-Memory (Zhang et al., 2025e)	Token-level	3D	QA	PE
OASIS (Yang et al., 2025)	Token-level, Parametric	1D	Social Simulation	PE

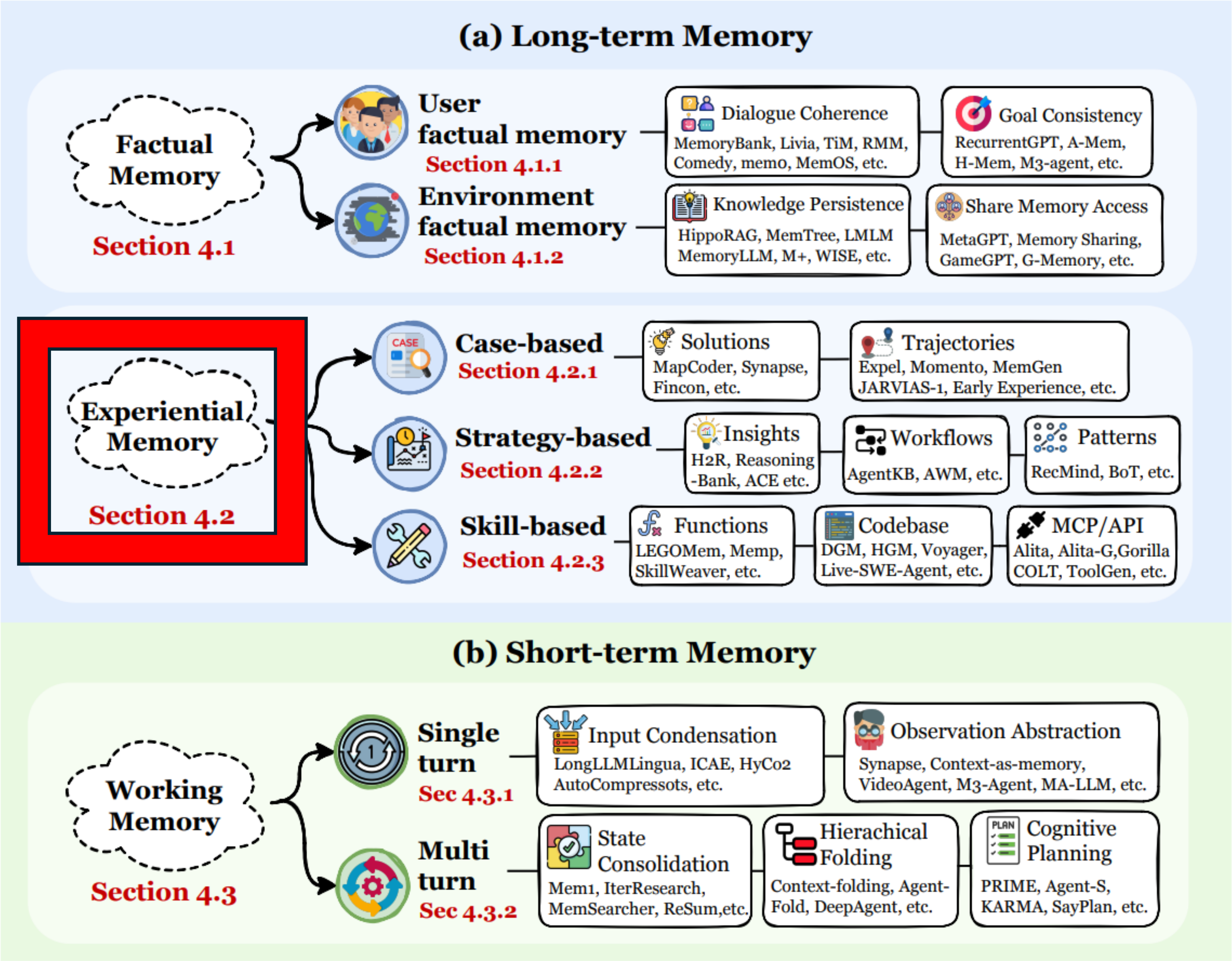


Figure 6 The functional taxonomy of agent memory. We organize memory capabilities based on their *functions* (purpose) into three primary pillars spanning two temporal domains: (1) **Factual Memory** serves as a persistent declarative knowledge base to ensure interaction *consistency*, *coherence*, and *adaptability*; (2) **Experiential Memory** encapsulates procedural knowledge to enable *continual learning* and *self-evolution* across episodes; and (3) **Working Memory** provides mechanisms for the active management of transient context.

Functions of Memory – Experiential

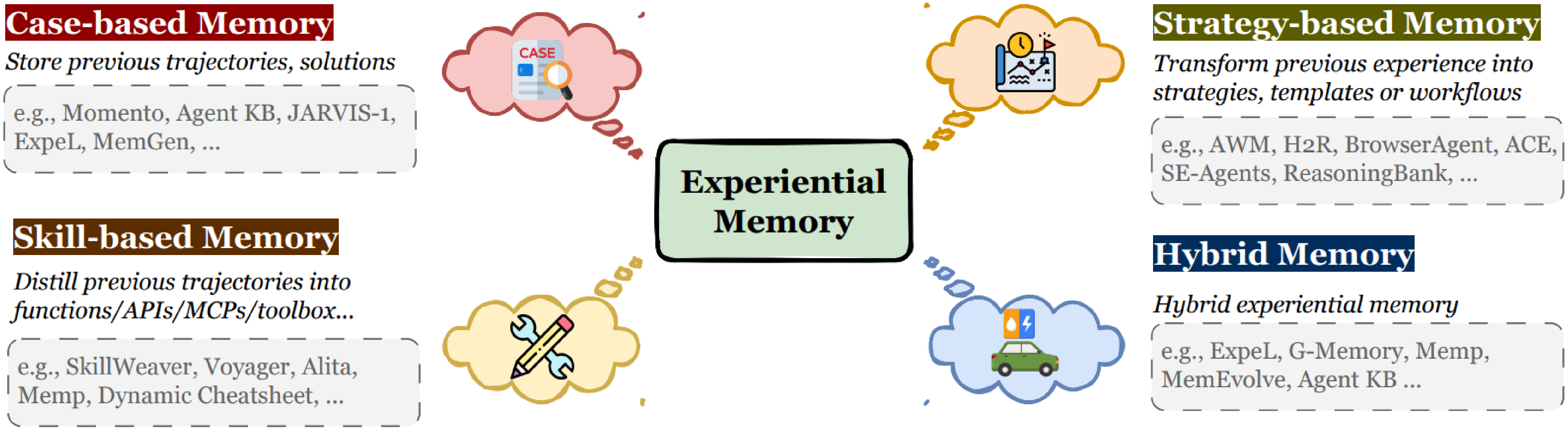


Figure 7 Taxonomy of experiential memory paradigms. We classify approaches based on the *abstraction level* of stored knowledge: (1) **Case-based Memory** preserves raw trajectories and solutions as concrete exemplars; (2) **Strategy-based Memory** abstracts experiences into high-level strategies, templates, or workflows; (3) **Skill-based Memory** distills procedural knowledge into executable functions and APIs; and (4) **Hybrid Memory** integrates multiple representations. Together, these systems mirror human *procedural memory* to enable continual learning and self-evolution. This figure draws inspiration from Gao et al. (2025).

Functions of Memory – Experiential

Three Types of Experiential Memory

- **Case-based Memory** (Section 4.2.1) stores minimally processed records of historical episodes, prioritizing high informational fidelity to support direct replay and imitation. By retaining the original alignment between situations and outcomes, it serves as a repository of concrete, verifiable evidence that functions as in-context exemplars for evidence-driven learning.
- **Strategy-based Memory** (Section 4.2.2) distills transferable reasoning patterns, workflows, and high-level insights from past trajectories to guide planning across diverse scenarios. Acting as a cognitive scaffold, it decouples decision-making logic from specific contexts, thereby enhancing cross-task generalization and constraining the search space for complex reasoning.
- **Skill-based Memory** (Section 4.2.3) encapsulates executable procedural capacities, ranging from atomic code snippets to standardized API protocols, that operationalize abstract strategies into verifiable actions. This category serves as the agent's active execution substrate, enabling the modular expansion of capabilities and the efficient handling of tool-use environments.

Table 5 Taxonomy of experiential memory methods. We categorize existing works based on the *abstraction level* of stored knowledge: **Case-based Memory** preserves raw records for direct replay, **Strategy-based Memory** distills abstract heuristics for planning, and **Skill-based Memory** compiles executable capabilities for action. Methods are compared across three technical dimensions: (1) **Carrier** (Section 3) identifies the storage medium, (2) **Form** specifies the representation format of the experience, and (3) **Optimization** denotes the integration strategy, where *PE* encompasses prompt engineering and inference-time techniques without parameter updates, distinct from gradient-based methods like *SFT* and *RL*.

Method	Carrier	Form	Task	Optimization
I. Case-based Memory				
Expel (Zhao et al., 2024)	Token-level	Solution	Reasoning	PE
Synapse (Zheng et al., 2024a)	Token-level	Solution	Web Interaction, Instruction-guided Web Task	PE
Fincon (Yu et al., 2024)	Token-level	Solution	Financial	PE
MapCoder (Islam et al., 2024)	Token-level	Solution	Coding	PE
Memento (Zhou et al., 2025a)	Token-level	Trajectory	Reasoning	RL
COLA (Zhao et al., 2025a)	Token-level	Trajectory	GUI, Web Navigation, Reasoning	PE
Continuous Memory (Wu et al., 2025e)	Latent	Trajectory	GUI	SFT
JARVIS-1 (Wang et al., 2025q)	Token-level	Trajectory	Game, GUI Interaction	PE
MemGen (Zhang et al., 2025d)	Latent	Trajectory	Web Search, Embodied Simulation, Reasoning, Math, Code	RL, SFT
Early Experience (Zhang et al., 2025k)	Parametric	Trajectory	Embodied Simulation, Reasoning, Web Navigation	SFT
DreamGym (Chen et al., 2025f)	Token-level	Trajectory	Web Interaction, Embodied Simulation, Shopping	RL
MemRL (Zhang et al., 2026)	Token-level	Trajectory	Coding, Embodied Simulation, Reasoning	RL
II. Strategy-based Memory				
Reflexion (Shinn et al., 2023a)	Token-level	Insight	Embodied Simulation, Reasoning, Coding	PE
Buffer of Thoughts (Yang et al., 2024b)	Token-level	Pattern	Game, Reasoning, Coding	PE
AWM (Wang et al., 2024m)	Token-level	Workflow	Web Interaction, Instruction-guided Web Task	PE
RecMind (Wang et al., 2024h)	Token-level	Pattern	Recommendation	PE
H^2R (Ye et al., 2025b)	Token-level	Insight	Game, Embodied Simulation	PE
ReasoningBank (Ouyang et al., 2025)	Token-level	Insight	Web Interaction, Instruction-guided Web Task	PE
R2D2 (Huang et al., 2025c)	Token-level	Insight	Web Interaction	PE

Continued on next page

Method	Carrier	Form	Task	Optimization
BrowserAgent (Yu et al., 2025d)	Token-level	Insight	General QA, Web search	RL, SFT
Agent KB (Tang et al., 2025d)	Token-level	Workflow	Code, Reasoning	PE
ToolMem (Xiao et al., 2025b)	Token-level	Insight	Reasoning, Image Generation	PE
PRINCIPLES (Kim et al., 2025a)	Token-level	Pattern	Emotional Companion	PE
SE-Agent (Sun et al., 2025c)	Token-level	Insight	Coding	PE
ACE (Zhang et al., 2025n)	Token-level	Insight	Coding, Tool calling, Financial	PE
Flex (Cai et al., 2025c)	Token-level	Insight	Math, Chemistry, Biology	PE
AgentEvolver (Zhai et al., 2025)	Parametric	Pattern	Tool-augmented Task	RL
Dynamic Cheatsheet (Suzgun et al., 2025)	Token-level	Insight	Math, Reasoning, Game	PE
Training-Free GRPO (Cai et al., 2025b)	Token-level	Insight	Math, Reasoning, Web Search	PE
MemEvolve (Zhang et al., 2025h)	Token-level	Solution,Insight	Web Search, Reasoning	PE
		III. Skill-based Memory		
CREATOR (Qian et al., 2023)	Token-level	Function and Script	Reasoning, Math	PE
Gorilla (Patil et al., 2024)	Token-level	API	Tool calling	SFT
ToolRerank (Zheng et al., 2024b)	Token-level	API	Tool calling	PE
Voyager (Wang et al., 2024b)	Token-level	Code Snippet	Game	PE
RepairAgent (Bouzenia et al., 2024)	Token-level	Function and Script	Coding	PE
COLT (Qu et al., 2024)	Token-level	API	Tool calling	SFT
ToolLLM (Qin et al., 2024a)	Token-level	API	Tool Calling	SFT
LEGOMem (Han et al., 2025a)	Token-level	Function and Script	Office	PE
Darwin Gödel Machine (Zhang et al., 2025i)	Token-level	Code Snippet	Code	PE
Huxley-Gödel Machine (Wang et al., 2025j)	Token-level	Code Snippet	Code	PE
Memp$^{\text{p}}$ (Fang et al., 2025d)	Token-level	Function and Script	Embodied Simulation, Travel Planning	PE
SkillWeaver (Zheng et al., 2025a)	Token-level	Function and Script	Web Interaction, Instruction-guided Web Task	PE
Alita (Qiu et al., 2025c)	Token-level	MCP	Math, Reasoning, VQA	PE
Alita-G (Qiu et al., 2025b)	Token-level	MCP	Math, Reasoning, VQA	PE
LearnAct (Liu et al., 2025b)	Token-level	Function and Script	Mobile GUI	PE
ToolGen (Wang et al., 2025i)	Parametric	API	Tool calling	SFT
MemTool (Lumer et al., 2025)	Token-level	MCP	Tool calling	SFT
ToolRet (Shi et al., 2025c)	Token-level	API	Web, Code, Tool Retrieval	SFT
DRAFT (Qu et al., 2025a)	Token-level	API	Tool calling	PE
ASI (Wang et al., 2025s)	Token-level	Functions and Scripts	Web Interaction	PE

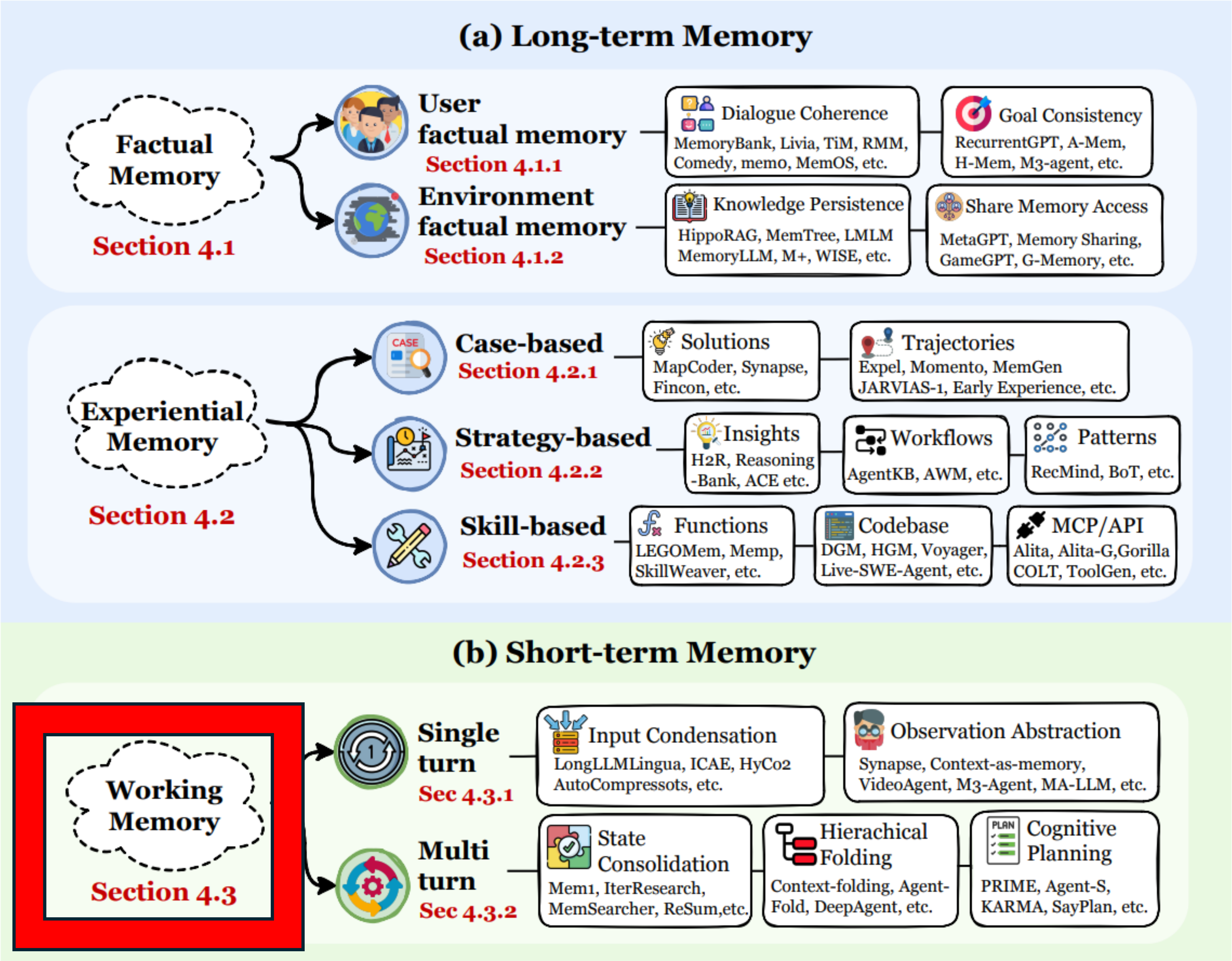


Figure 6 The functional taxonomy of agent memory. We organize memory capabilities based on their *functions* (purpose) into three primary pillars spanning two temporal domains: (1) **Factual Memory** serves as a persistent declarative knowledge base to ensure interaction *consistency*, *coherence*, and *adaptability*; (2) **Experiential Memory** encapsulates procedural knowledge to enable *continual learning* and *self-evolution* across episodes; and (3) **Working Memory** provides mechanisms for the active management of transient context.

Functions of Memory -- Working

Two Types of Working Memory

- **Single-turn Working Memory** (Section 4.3.1) focuses on **input condensation and abstraction**. In this setting, the system must process massive immediate inputs such as long documents or high-dimensional multimodal streams within a single forward pass. The goal is to dynamically filter and rewrite evidence to construct a bounded computational scratchpad, thereby maximizing the effective information payload per token.

- **Multi-turn Working Memory** (Section 4.3.2) addresses **temporal state maintenance**. In sequential interactions, the challenge is to prevent historical accumulation from overwhelming the attention mechanism. This involves maintaining task states, goals, and constraints through a continuous loop of reading, executing, and updating, ensuring that intermediate artifacts are folded and consolidated across turns.

Method	Carrier	Task	Optimization
I. Single-turn Working Memory			
(a) Input Condensation			
Gist (Mu et al., 2023)	Latent	Instruction Fine-tuning	SFT
ICAE (Ge et al., 2024)	Latent	Language Modeling, Instruction Fine-tuning	Pretrain, LoRA
AutoCompressors (Chevalier et al., 2023)	Latent	Langague Modeling	SFT
LLMLingua (Jiang et al., 2023)	Token-level	Reasoning, Conversation, Summarization	PE
LongLLMLingua (Jiang et al., 2024)	Token-level	Multi-doc QA, Long-context, Multi-hop QA	PE
CompAct (Yoon et al., 2024)	Token-level	Document QA	SFT
HyCo2 (Liao et al., 2025a)	Hybrid	Summarization, Open-domain QA, Multi-hop QA	SFT
Sentence-Anchor (Tarasov et al., 2025)	Latent	Document QA	SFT
MELODI (Chen et al., 2024c)	Hybrid	Pretraining	Pretrain
R^3Mem (Wang et al., 2025k)	Latent	Document QA, Language Modeling	PEFT
(b) Observation Abstraction			
Synapse (Zheng et al., 2024a)	Token-level	Computer Control, Web Navigation	PE
VideoAgent (Wang et al., 2024g)	Token-level	Long-term Video Understanding	PE
MA-LMM (He et al., 2024)	Latent	Long-term Video Understanding	SFT
Context as Memory (Yu et al., 2025b)	Token-level	Long-term Video Generation	PE
II. Multi-turn Working Memory			
(c) State Consolidation			
MEM1 (Zhou et al., 2025b)	Latent	Retrieval, Open-domain QA, Shopping	RL
MemGen (Zhang et al., 2025d)	Latent	Reasoning, Embodied Action, Web Search, Coding	RL
MemAgent (Yu et al., 2025a)	Token-level	Long-term Doc. QA	RL
ReMemAgent (Shi et al., 2025b)	Token-level	Long-term Doc. QA	RL
ReSum (Wu et al., 2025f)	Token-level	Long-horizon Web Search	RL
MemSearcher (Yuan et al., 2025a)	Token-level	Multi-hop QA	SFT, RL
ACON (Kang et al., 2025c)	Token-level	App use, Multi-objective QA	PE
IterResearch (Chen et al., 2025b)	Token-level	Reasoning, Web Navigation, Long-Horizon QA	RL
SUPO (Lu et al., 2025a)	Token-level	Long-horizon task	RL
AgentDiet (Xiao et al., 2025a)	Token-level	Long-horizon task	PE
SUMER (Zheng et al., 2025c)	Token-level	QA	RL
Sculptor (Li et al., 2025f)	Token-level	Multi-Needle QA	PE,RL
AgeMem (Yu et al., 2026)	Token-level	QA, Embodied Action	PE,RL
(d) Hierarchical Folding			
HiAgent (Hu et al., 2025a)	Token-level	Long-horizon Agent Task	PE
Context-Folding (Sun et al., 2025b)	Token-level	Deep Research, SWE	RL
AgentFold (Ye et al., 2025a)	Token-level	Web Search	SFT
DeepAgent (Li et al., 2025i)	Token-level	Tool Use, Shopping, Reasoning	RL
(e) Cognitive Planning			
SayPlan (Rana et al., 2023)	Token-level	3D Scene Graph, Robotics	PE
KARMA (Wang et al., 2025r)	Token-level	Household	PE
Agent-S (Agashe et al., 2025)	Token-level	Computer Use	PE
PRIME (Tran et al., 2025)	Token-level	Multi-hop QA, Knowledge-intensive Reasoning	PE

University of Virginia

Table 6 Taxonomy of working memory methods. We categorize approaches into **Single-turn** and **Multi-turn** settings based on interaction dynamics. Methods are compared across three technical dimensions: (1) **Carrier** (Section 3) identifies the storage medium, (2) **Task** specifies the evaluation domain or application scenario, and (3) **Optimization** denotes the integration strategy, where PE encompasses prompt engineering and inference-time techniques without parameter updates, distinct from gradient-based methods like SFT and RL.

Memory Dynamics

UNIVERSITY of VIRGINIA

Three Fundamental Process in Memory Systems

1. **Memory Formation** (Section 5.1): This process focuses on transforming raw experience into information-dense knowledge. Instead of passively logging all interaction history, the memory system selectively identifies information with long-term utility, such as successful reasoning patterns or environmental constraints. This part answers the question: "How to extract the memory?".
2. **Memory Evolution** (Section 5.2): This process represents the dynamic evolution of the memory system. It focuses on integrating newly formed memories with the existing memory base. Through mechanisms such as the consolidation of correlated entries, conflict resolution, and adaptive pruning, the system ensures that the memory remains generalizable, coherent, and efficient in an ever-changing environment. This part answers the question: "How to refine the memory?".
3. **Memory Retrieval** (Section 5.3): This process determines the quality of the retrieved memory. Conditioned on the context, the system constructs a task-aware query and uses a carefully designed retrieval strategy to access the appropriate memory bank. The retrieved memory is therefore both semantically relevant and functionally critical for reasoning. This part answers the question: "How to utilize the memory?".

Memory Dynamics

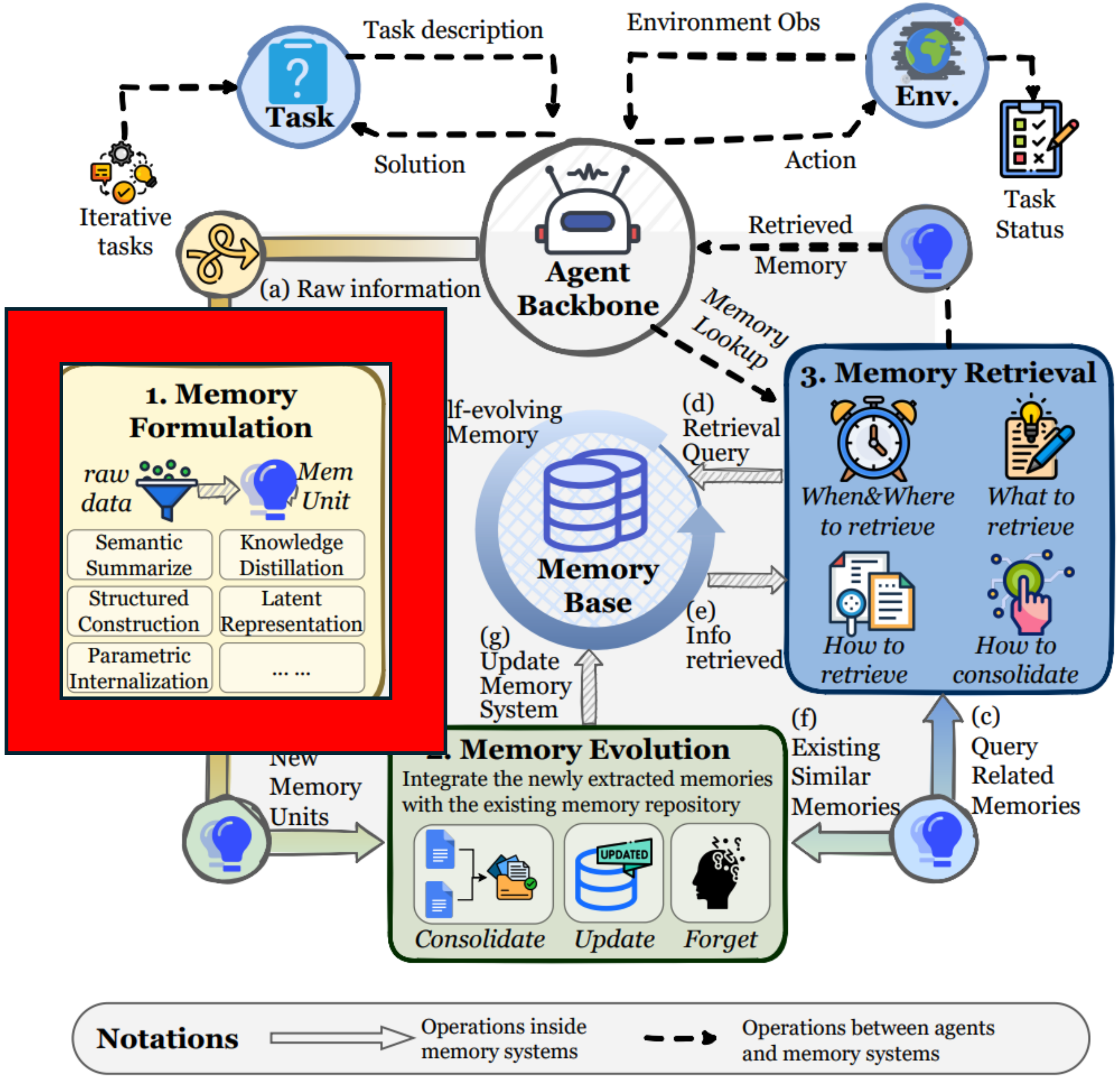


Figure 8 The operational dynamics of agent memory. We decouple the complete memory lifecycle into three fundamental processes that drive the system's adaptability and self-evolution: **(1) Memory Formation** transforms raw interactive experiences into information-dense knowledge units by selectively identifying patterns with long-term utility; **(2) Memory Evolution** dynamically integrates new memories into the existing repository through consolidation, updating, and forgetting mechanisms to ensure the knowledge base remains coherent and efficient; and **(3) Memory Retrieval** executes context-aware queries to access specific memory modules, thereby optimizing reasoning performance with precise information support. The alphabetical order denotes the sequence of operations within the memory systems.

Memory Dynamics -- Formation

University of Virginia

Five Categories of Memory Formation Operations

- **Semantic Summarization**(Section 5.1.1) transforms lengthy raw data into compact summaries, filtering out redundancy while preserving global, high-level semantic information to reduce contextual overhead.
- **Knowledge Distillation** (Section 5.1.2) extracts specific cognitive assets, ranging from factual details to experiential planning strategies.
- **Structured Construction**(Section 5.1.3) organizes amorphous source data into explicit topological representations, such as knowledge graphs or hierarchical trees, to enhance the explainability of memory and support multi-hop reasoning.
- **Latent Representation**(Section 5.1.4) encodes raw experiences directly into machine-native formats (e.g., vector embeddings or KV states) within a continuous latent space.
- **Parametric Internalization**(Section 5.1.5) consolidates external memories directly into the model's weight space through parameter updates, effectively transforming retrievable information into the agent's intrinsic competence and instincts.

Method	Sub-Type	Representation Form	Key Mechanism
I. Semantic Summarization			
MemGPT (Packer et al., 2023a)	Incremental	Textual Summary	Merging new chunks into the working context
Mem0 (Chhikara et al., 2025)	Incremental	Textual Summary	LLM-driven summarization
Mem1 (Zhou et al., 2025b)	Incremental	Textual Summary	RL-optimized summarization (PPO)
MemAgent (Yu et al., 2025a)	Incremental	Textual Summary	RL-optimized summarization (GRPO)
MemoryBank (Zhong et al., 2024)	Partitioned	Textual Summary	Daily/Session-based segmentation
ReadAgent (Lee et al., 2024a)	Partitioned	Textual Summary	Semantic clustering before summarization
LightMem (Fang et al., 2025b)	Partitioned	Textual Summary	Topic-clustered summarization
DeepSeek-OCR (Wei et al., 2025a)	Partitioned	Visual Token Mapping	Optical 2D mapping compression
FDVS (You et al., 2024)	Partitioned	Multimodal Summary	Multi-source signal integration (Subtitle/Object)
LangRepo (Kahatapitiya et al., 2025)	Partitioned	Multimodal Summary	Hierarchical video clip aggregation
II. Knowledge Distillation			
TiM (Liu et al., 2023a)	Factual	Textual Insight	Abstraction of dialogue into thoughts
RMM (Tan et al., 2025b)	Factual	Topic Insight	Abstraction of dialogue into topic-based memory
MemGuide (Du et al., 2025b)	Factual	User Intent	Capturing high-level user intent
M3-Agent (Long et al., 2025)	Factual	Text-addressable Facts	Compressing egocentric visual observations
AWM (Wang et al., 2024m)	Experiential	Workflow Patterns	Workflow extraction from success trajectories
Mem^p (Fang et al., 2025d)	Experiential	Procedural Knowledge	Distilling gold trajectories into abstract procedures
ExpeL (Zhao et al., 2024)	Experiential	Experience Insight	Contrastive reflection and successful practices
R2D2 (Huang et al., 2025c)	Experiential	Reflective Insight	Reflection on reasoning traces vs. ground truth
H^2R (Ye et al., 2025b)	Experiential	Hierarchical Insight	Two-tier reflection (Plan & Subgoal)
Memory-R1 (Yan et al., 2025c)	Experiential	Textual Knowledge	RL-trained LLMExtract module
Mem-α (Wang et al., 2025p)	Experiential	Textual Insight	Learnable insight extraction policy
III. Structured Construction			
KGT (Sun et al., 2024)	Entity-Level	User Graph	Encoding user preferences as nodes/edges
$Mem0^g$ (Chhikara et al., 2025)	Entity-Level	Knowledge Graph	LLM-based entity and triplet extraction
D-SMART (Lei et al., 2025)	Entity-Level	Dynamic Memory Graph	Constructing an OWL-compliant graph
GraphRAG (Edge et al., 2025)	Entity-Level	Hierarchical KG	Community detection and iterative summarization
AriGraph (Anokhin et al., 2024)	Entity-Level	Semantic+Episodic Graph	Dual-layer (Semantic nodes + Episodic links)
Zep (Rasmussen et al., 2025)	Entity-Level	Temporal KG	3-layer graph (Episodic, Semantic, Community)
RAPTOR (Sarthi et al., 2024)	Chunk-Level	Tree Structure	Recursive GMM clustering and summarization
MemTree (Rezazadeh et al., 2025c)	Chunk-Level	Tree Structure	Bottom-up insertion and summary updates
H-MEM (Sun and Zeng, 2025)	Chunk-Level	Hierarchical JSON	Top-down 4-level hierarchy organization
A-MEM (Xu et al., 2025c)	Chunk-Level	Networked Notes	Discrete notes with semantic links
PREMem (Kim et al., 2025b)	Chunk-Level	Reasoning Patterns	Cross-session reasoning pattern clustering
CAM (Li et al., 2025g)	Chunk-Level	Hierarchical Graph	Disentangling overlapping clusters via replication
G-Memory (Zhang et al., 2025c)	Chunk-Level	Hierarchical Graph	3-tier graph (interaction, query, insight)
IV. Latent Representation			
MemoryLLM (Wang et al., 2024j)	Textual	Latent Vector	Self-updatable latent embeddings
M+ (Wang et al., 2025n)	Textual	Latent Vector	Cross-layer long-term memory tokens
MemGen (Zhang et al., 2025d)	Textual	Latent Token	Latent memory trigger and weaver
ESR (Shen et al., 2024)	Multimodal	Latent Vector	Video-to-Language-to-Vector encoding
CoMEM (Wu et al., 2025d)	Multimodal	Continuous Embedding	Vision-language compression via Q-Former
Mem2Ego (Zhang et al., 2025m)	Multimodal	Multimodal Embedding	Embedding landmark semantics as latent memory
KARMA (Wang et al., 2025r)	Multimodal	Multimodal Embedding	Hybrid long/short-term memory encoding
V. Parametric Internalization			
MEND (Mitchell et al., 2022)	Knowledge	Gradient Decomposition	Auxiliary network for fast edits
ROME (Meng et al., 2022)	Knowledge	Model Parameters	Causal tracing and rank-one update
MEMIT (Meng et al., 2023)	Knowledge	Model Parameters	Mass-editing via residual distribution
CoLoR (Wistuba et al., 2023)	Knowledge	LoRA Parameters	Low-rank adapter training
ToolFormer (Schick et al., 2023)	Capability	Model Parameters	Supervised fine-tuning on API calls

Table 7 Taxonomy of memory formation methods. We classify approaches based on the memory formation operations. Methods are analyzed across three technical dimensions: (1) **Sub-Type** identifies the specific variation or scope, (2) **Representation Form** specifies the output format, and (3) **Key Mechanism** denotes the core algorithmic strategy.

- **Semantic Summarization**(Section 5.1.1) transforms lengthy raw data into compact summaries, filtering out redundancy while preserving global, high-level semantic information to reduce contextual overhead.

Method	Sub-Type	Representation Form	Key Mechanism
		I. Semantic Summarization	
MemGPT (Packer et al., 2023a)	Incremental	Textual Summary	Merging new chunks into the working context
Mem0 (Chhikara et al., 2025)	Incremental	Textual Summary	LLM-driven summarization
Mem1 (Zhou et al., 2025b)	Incremental	Textual Summary	RL-optimized summarization (PPO)
MemAgent (Yu et al., 2025a)	Incremental	Textual Summary	RL-optimized summarization (GRPO)
MemoryBank (Zhong et al., 2024)	Partitioned	Textual Summary	Daily/Session-based segmentation
ReadAgent (Lee et al., 2024a)	Partitioned	Textual Summary	Semantic clustering before summarization
LightMem (Fang et al., 2025b)	Partitioned	Textual Summary	Topic-clustered summarization
DeepSeek-OCR (Wei et al., 2025a)	Partitioned	Visual Token Mapping	Optical 2D mapping compression
FDVS (You et al., 2024)	Partitioned	Multimodal Summary	Multi-source signal integration (Subtitle/Object)
LangRepo (Kahatapitiya et al., 2025)	Partitioned	Multimodal Summary	Hierarchical video clip aggregation

- **Knowledge Distillation** (Section 5.1.2) extracts specific cognitive assets, ranging from factual details to experiential planning strategies.

II. Knowledge Distillation

TiM (Liu et al., 2023a)	Factual	Textual Insight	Abstraction of dialogue into thoughts
RMM (Tan et al., 2025b)	Factual	Topic Insight	Abstraction of dialogue into topic-based memory
MemGuide (Du et al., 2025b)	Factual	User Intent	Capturing high-level user intent
M3-Agent (Long et al., 2025)	Factual	Text-addressable Facts	Compressing egocentric visual observations
AWM (Wang et al., 2024m)	Experiential	Workflow Patterns	Workflow extraction from success trajectories
Mem^p (Fang et al., 2025d)	Experiential	Procedural Knowledge	Distilling gold trajectories into abstract procedures
ExpeL (Zhao et al., 2024)	Experiential	Experience Insight	Contrastive reflection and successful practices
R2D2 (Huang et al., 2025c)	Experiential	Reflective Insight	Reflection on reasoning traces vs. ground truth
H^2R (Ye et al., 2025b)	Experiential	Hierarchical Insight	Two-tier reflection (Plan & Subgoal)
Memory-R1 (Yan et al., 2025c)	Experiential	Textual Knowledge	RL-trained LLMExtract module
Mem-α (Wang et al., 2025p)	Experiential	Textual Insight	Learnable insight extraction policy

- **Structured Construction**(Section 5.1.3) organizes amorphous source data into explicit topological representations, such as knowledge graphs or hierarchical trees, to enhance the explainability of memory and support multi-hop reasoning.

III. Structured Construction

KGT (Sun et al., 2024)	Entity-Level	User Graph	Encoding user preferences as nodes/edges
Mem0^g (Chhikara et al., 2025)	Entity-Level	Knowledge Graph	LLM-based entity and triplet extraction
D-SMART (Lei et al., 2025)	Entity-Level	Dynamic Memory Graph	Constructing an OWL-compliant graph
GraphRAG (Edge et al., 2025)	Entity-Level	Hierarchical KG	Community detection and iterative summarization
AriGraph (Anokhin et al., 2024)	Entity-Level	Semantic+Episodic Graph	Dual-layer (Semantic nodes + Episodic links)
Zep (Rasmussen et al., 2025)	Entity-Level	Temporal KG	3-layer graph (Episodic, Semantic, Community)
RAPTOR (Sarthi et al., 2024)	Chunk-Level	Tree Structure	Recursive GMM clustering and summarization
MemTree (Rezazadeh et al., 2025c)	Chunk-Level	Tree Structure	Bottom-up insertion and summary updates
H-MEM (Sun and Zeng, 2025)	Chunk-Level	Hierarchical JSON	Top-down 4-level hierarchy organization
A-MEM (Xu et al., 2025c)	Chunk-Level	Networked Notes	Discrete notes with semantic links
PREMem (Kim et al., 2025b)	Chunk-Level	Reasoning Patterns	Cross-session reasoning pattern clustering
CAM (Li et al., 2025g)	Chunk-Level	Hierarchical Graph	Disentangling overlapping clusters via replication
G-Memory (Zhang et al., 2025c)	Chunk-Level	Hierarchical Graph	3-tier graph (interaction, query, insight)

- **Latent Representation**(Section 5.1.4) encodes raw experiences directly into machine-native formats (e.g., vector embeddings or KV states) within a continuous latent space.

IV. Latent Representation

MemoryLLM (Wang et al., 2024j)	Textual	Latent Vector	Self-updatable latent embeddings
M+ (Wang et al., 2025n)	Textual	Latent Vector	Cross-layer long-term memory tokens
MemGen (Zhang et al., 2025d)	Textual	Latent Token	Latent memory trigger and weaver
ESR (Shen et al., 2024)	Multimodal	Latent Vector	Video-to-Language-to-Vector encoding
CoMEM (Wu et al., 2025d)	Multimodal	Continuous Embedding	Vision-language compression via Q-Former
Mem2Ego (Zhang et al., 2025m)	Multimodal	Multimodal Embedding	Embedding landmark semantics as latent memory
KARMA (Wang et al., 2025r)	Multimodal	Multimodal Embedding	Hybrid long/short-term memory encoding

- **Parametric Internalization**(Section 5.1.5) consolidates external memories directly into the model's weight space through parameter updates, effectively transforming retrievable information into the agent's intrinsic competence and instincts.

V. Parametric Internalization

MEND (Mitchell et al., 2022)	Knowledge	Gradient Decomposition	Auxiliary network for fast edits
ROME (Meng et al., 2022)	Knowledge	Model Parameters	Causal tracing and rank-one update
MEMIT (Meng et al., 2023)	Knowledge	Model Parameters	Mass-editing via residual distribution
CoLoR (Wistuba et al., 2023)	Knowledge	LoRA Parameters	Low-rank adapter training
ToolFormer (Schick et al., 2023)	Capability	Model Parameters	Supervised fine-tuning on API calls

Memory Dynamics

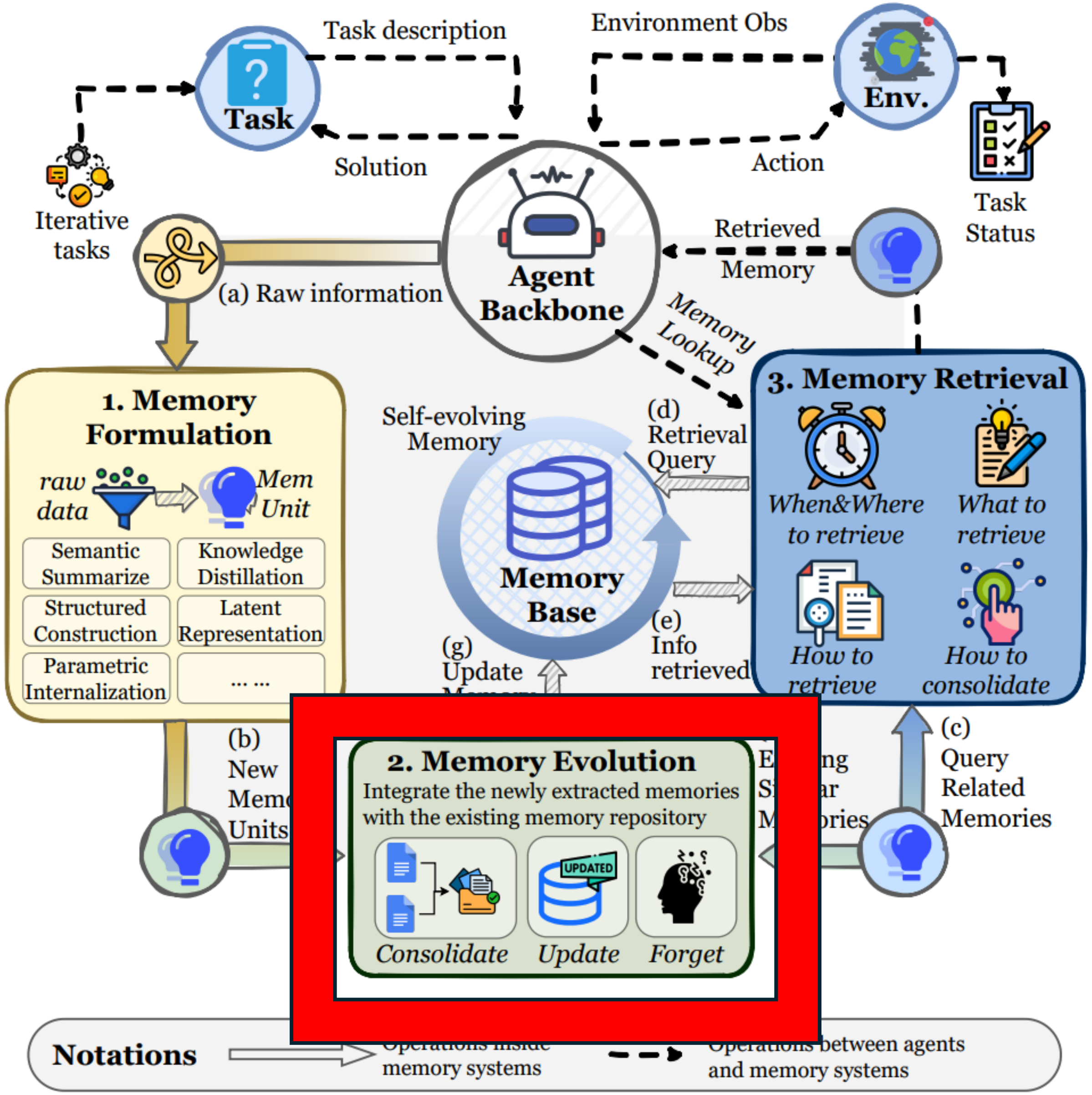


Figure 8 The operational dynamics of agent memory. We decouple the complete memory lifecycle into three fundamental processes that drive the system's adaptability and self-evolution: **(1) Memory Formation** transforms raw interactive experiences into information-dense knowledge units by selectively identifying patterns with long-term utility; **(2) Memory Evolution** dynamically integrates new memories into the existing repository through consolidation, updating, and forgetting mechanisms to ensure the knowledge base remains coherent and efficient; and **(3) Memory Retrieval** executes context-aware queries to access specific memory modules, thereby optimizing reasoning performance with precise information support. The alphabetical order denotes the sequence of operations within the memory systems.

Memory Dynamics -- Evolution

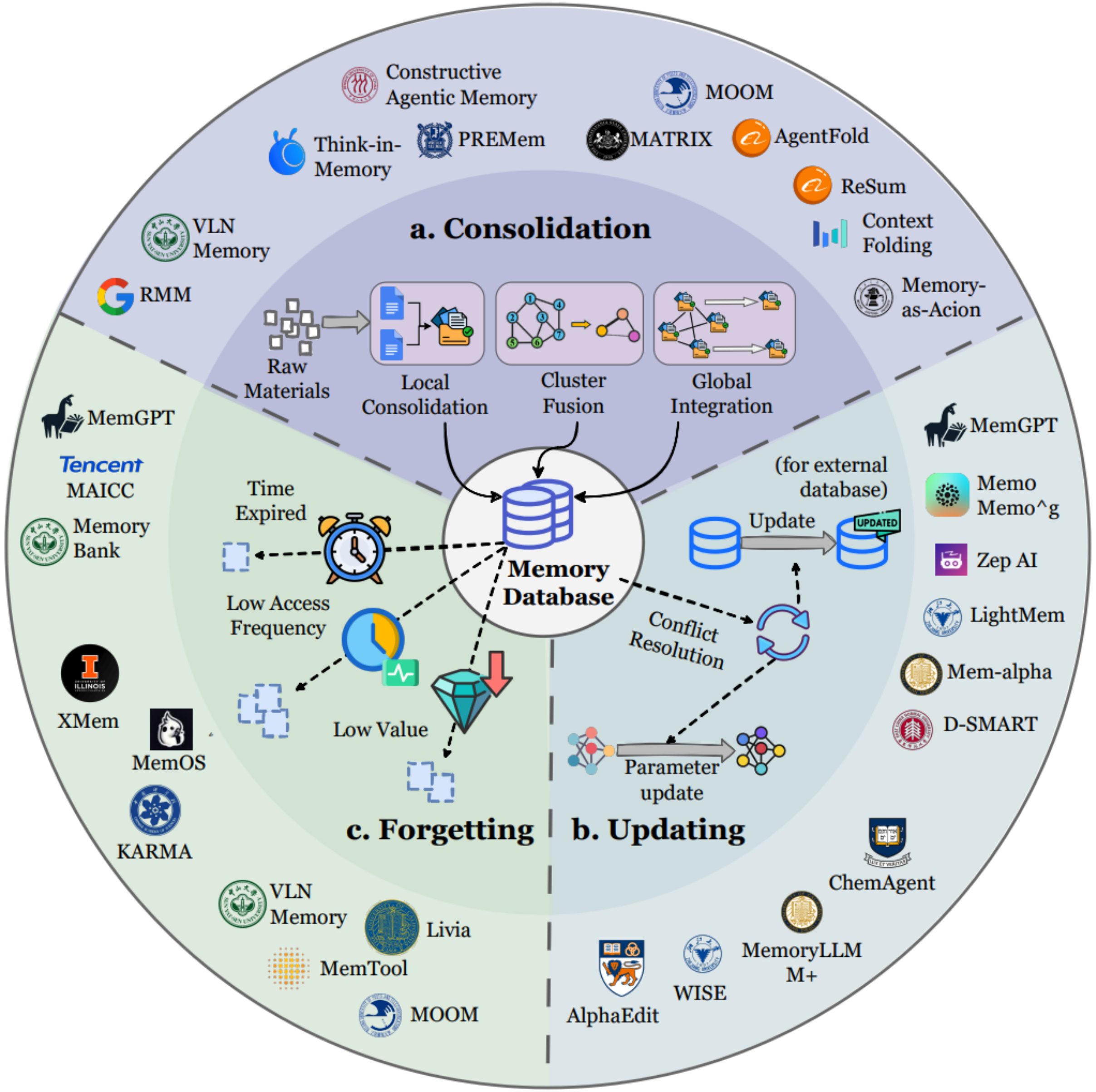


Figure 9 The landscape of Memory Evolution mechanisms. We categorize the evolution process into three distinct branches that maintain the central Memory Database: **(a) Consolidation** synthesizes insights by processing raw materials through local consolidation, cluster fusion, and global integration; **(b) Updating** ensures accuracy and consistency by performing conflict resolution on external databases and applying parameter updates to the internal model; and **(c) Forgetting** optimizes efficiency by pruning data based on specific criteria: time expiration, low access frequency, and low informational value. The outer ring displays representative frameworks and agents associated with each evolutionary mechanism.

Memory Dynamics -- Evolution

Three Mechanisms of Memory Evolution

- **Memory Consolidation** (Section 5.2.1) merges new and existing memories and performs reflective integration, forming more generalized insights. This ensures that learning is cumulative rather than isolated.
- **Memory Updating** (Section 5.2.2) resolves conflicts between new and existing memories, correcting and supplementing the repository to maintain accuracy and relevance. It allows the agent to adapt to changes in the environment or task requirements.
- **Memory Forgetting** (Section 5.2.3) removes outdated or redundant information, freeing capacity and improving efficiency. This prevents performance degradation due to knowledge overload and ensures that the memory repository remains focused on actionable and current knowledge.

Memory Dynamics

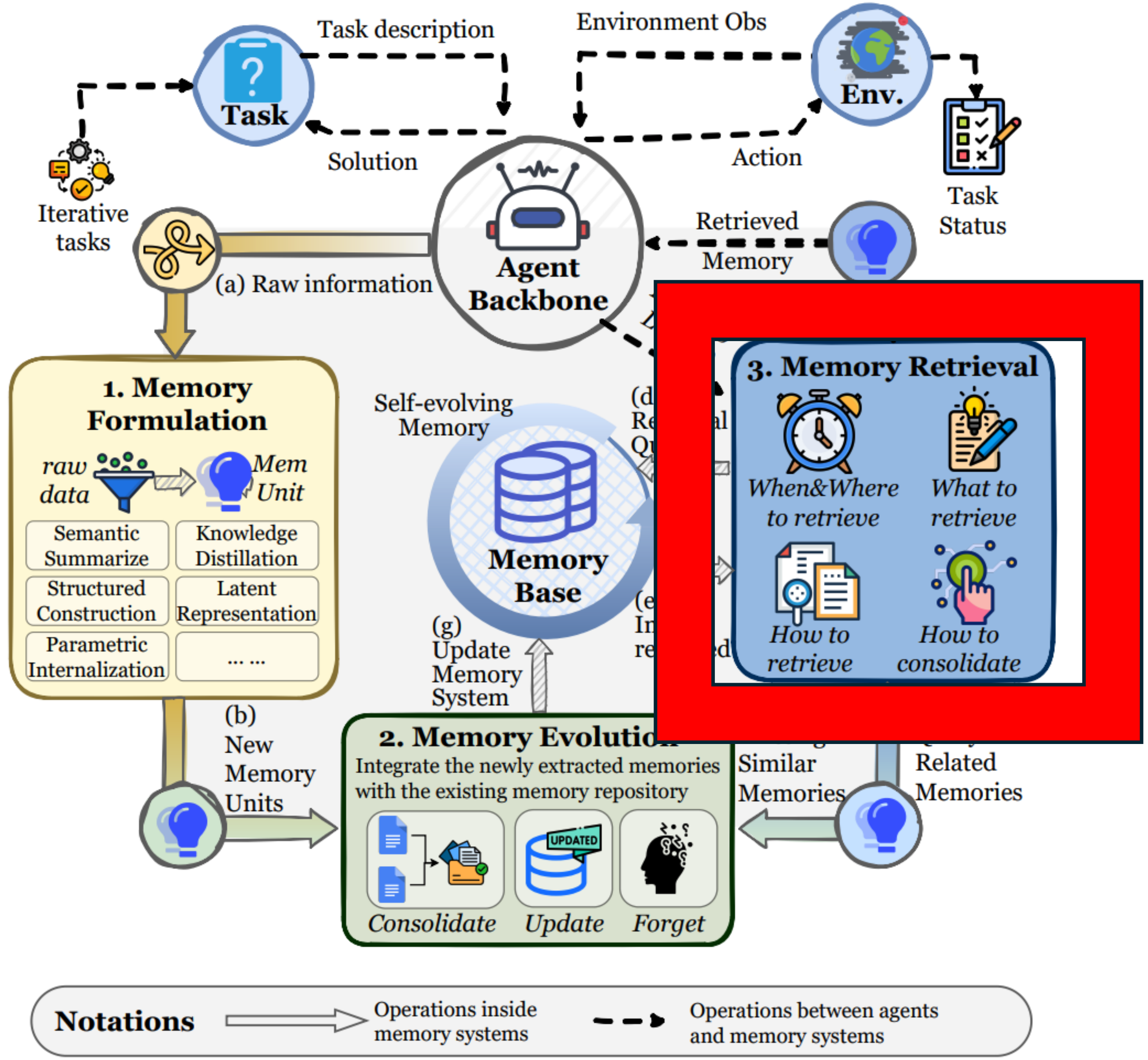


Figure 8 The operational dynamics of agent memory. We decouple the complete memory lifecycle into three fundamental processes that drive the system's adaptability and self-evolution: **(1) Memory Formation** transforms raw interactive experiences into information-dense knowledge units by selectively identifying patterns with long-term utility; **(2) Memory Evolution** dynamically integrates new memories into the existing repository through consolidation, updating, and forgetting mechanisms to ensure the knowledge base remains coherent and efficient; and **(3) Memory Retrieval** executes context-aware queries to access specific memory modules, thereby optimizing reasoning performance with precise information support. The alphabetical order denotes the sequence of operations within the memory systems.

Memory Dynamics -- Retrieval

Four Steps of Memory Retrieval

- **Retrieval Timing and Intent** (Section 5.3.1) determines the specific moments and objectives for memory retrieval, shifting from passive, instruction-driven triggers to autonomous, self-regulated decisions.
- **Query Construction** (Section 5.3.2) bridges the semantic gap between the user's raw input and the stored memory index by decomposing or rewriting queries into effective retrieval signals.
- **Retrieval Strategies** (Section 5.3.3) executes the search over the memory repository, employing paradigms ranging from sparse lexical matching to dense semantic embedding and structure-aware graph traversal.
- **Post-Retrieval Processing** (Section 5.3.4) refines the retrieved raw fragments through re-ranking, filtering, and aggregation, ensuring that the final context provided to the model is concise and coherent.

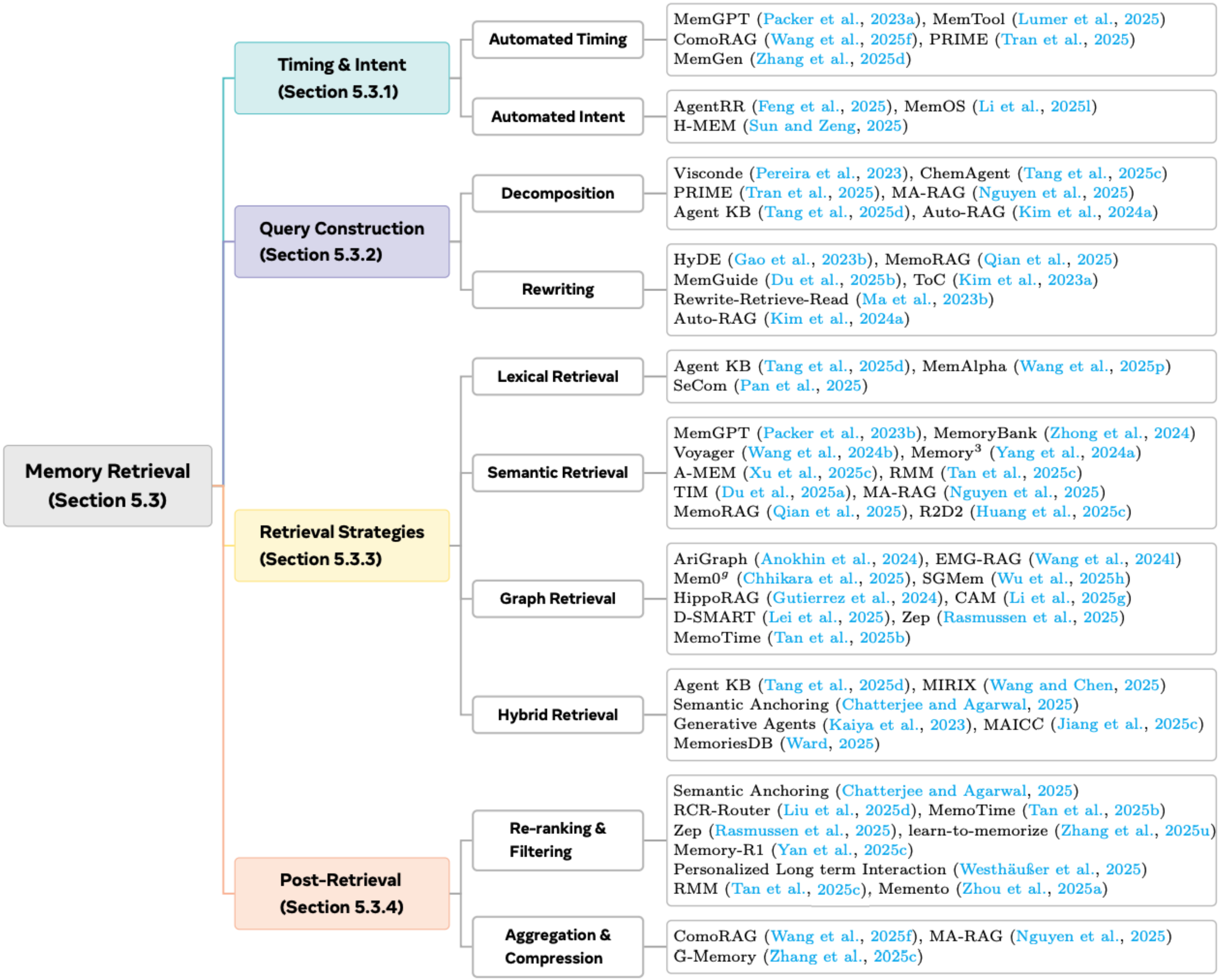


Figure 10 Taxonomy of memory retrieval methodologies in agentic systems. The mindmap organizes existing literature into four distinct phases of the retrieval pipeline: **Timing and Intent**, which governs the initiation of the process; Query Construction, covering techniques for query decomposition and rewriting; **Retrieval Strategies**, categorizing search paradigms into lexical, semantic, graph-based, and hybrid approaches; and **Post-Retrieval Processing**, which focuses on refining outputs through re-ranking, filtering, and aggregation.

Resources & Benchmarks

UNIVERSITY of VIRGINIA

Name	Link	Fac.	Exp.	MM.	Env.	Feature	Scale
Memory/Lifelong-learning/Self-evolving-oriented Benchmarks							
MemBench	GitHub	✔	✔	✘	simulated	interactive scenarios	53,000 s.
MemoryAgentBench	GitHub	✔	✔	✘	simulated	multi-turn interactions	4 t.
LoCoMo	Website	✔	✘	✔	real	conversational memory	300 s.
WebChoreArena	GitHub	✔	✔	✔	real	tedious web browsing	4 t./532 s.
MT-Mind2Web	GitHub	✔	✔	✘	real	conversational web navigation	720 s.
PersonaMem	Website	✔	✘	✘	simulated	dynamic user profiling	15 t./180 s.
LongMemEval	GitHub	✔	✘	✘	simulated	interactive memory	5 t./500 s.
PerLTQA	Website	✔	✘	✘	simulated	social personalized interactions	8,593 s.
MemoryBank	Website	✔	✘	✘	simulated	user memory updating	194 s.
MPR	GitHub	✔	✘	✘	simulated	user personalization	108,000 s.
PrefEval	Website	✔	✘	✘	simulated	personal preferences	3,000 s.
LOCCO	Website	✔	✘	✘	simulated	chronological conversations	3,080 s.
StoryBench	Website	✔	✔	✘	mixed	interactive fiction games	3 t.
MemoryBench	Website	✔	✔	✘	simulated	continual learning	4 t./$\sim$ 20,000 s.
Madial-Bench	GitHub	✔	✘	✘	simulated	memory recalling	331 s.
Evo-Memory	Website	✔	✔	✘	simulated	test-time learning	10 t./$\sim$ 3,700 s.
LifelongAgentBench	Website	✔	✔	✘	simulated	lifelong learning	1,396 s.
StreamBench	Website	✔	✔	✘	simulated	continuous online learning	9,702 s.
DialSim	Website	✔	✔	✘	real	multi-dialogue understanding	$\sim$ 1,300 s.
LongBench	Website	✔	✘	✘	mixed	long-context understanding	21 t./4,750 s.
LongBench v2	Website	✔	✘	✘	mixed	long-context multitasks	20 t./503 s.
RULER	GitHub	✔	✘	✘	simulated	long-context retrieval	13 t.
BABILong	GitHub	✔	✘	✘	simulated	long-context reasoning	20 t.
MM-Needle	Website	✔	✘	✔	simulated	multimodal long-context retrieval	$\sim$ 280,000 s.
HaluMem	GitHub	✔	✘	✘	simulated	memory hallucinations	3,467 s.
HotpotQA	Website	✔	✘	✘	simulated	long-context QA	113k s.
Other Related Benchmarks							
ALFWorld	Website	✔	✔	✘	simulated	text-based embodied environment	3,353 t.
ScienceWorld	GitHub	✔	✔	✘	simulated	interactive embodied environment	10 t./30 t.
AgentGym	Website	✘	✔	✘	mixed	multiple environments	89 t./20,509 s.
AgentBoard	GitHub	✘	✔	✘	mixed	multi-round interaction	9 t./1013 s.
PDDL*	Website	✘	✔	✘	simulated	strategy game	-
BabyAI	Website	✘	✔	✘	simulated	language learning	19 t.
WebShop	Website	✘	✔	✔	simulated	e-commerce web interaction	12,087 s.
WebArena	Website	✘	✔	✔	real	web interaction	812 s.
MMInA	Website	✔	✔	✔	real	multihop web interaction	1,050 s.
SWE-Bench Verified	Website	✘	✔	✘	real	code repair	500 s.
GAIA	Website	✘	✔	✔	real	human-level deep research	466 s.
xBench-DS	Website	✘	✔	✔	real	deep-search evaluation	100 s.
ToolBench	GitHub	✘	✔	✘	real	API tool use	126,486 s.
GenAI-Bench	Website	✘	✔	✔	real	visual generation evaluation	$\sim$ 40,000 s.

Table 8 Overview of benchmarks relevant to LLM agent memory, long-term, lifelong learning, and self-evolving evaluation. The table covers two categories of benchmarks: (i) benchmarks explicitly designed for memory-, lifelong learning-, or self-evolving agent evaluation, and (ii) other agent-oriented benchmarks that implicitly stress long-horizon memory through sequential, multi-step, or multi-task interactions. **Fac.** and **Exp.** indicate whether a benchmark evaluates factual memory or experiential (interaction-derived) memory, respectively. **MM.** denotes the presence of multimodal inputs, while **Env.** indicates whether the benchmark is conducted in a simulated or real environment. **Feature** summarizes the primary capability under evaluation, and **Scale** reports the approximate benchmark size in terms of samples (s.) or tasks (t.). PDDL denotes commonly used PDDL-based planning subsets.

Benchmarks & Dataset

Framework	Links	Fac.	Exp.	MM.	Structure	Evaluation
MemGPT	GitHub Website	✔	✔	✘	hierachical (S/LTM)	LoCoMo
Mem0	GitHub Website	✔	✔	✘	graph + vector	LoCoMo
Memobase	GitHub Website	✔	✔	✘	structured profiles	LoCoMo
MIRIX	GitHub Website	✔	✔	✔	structured memory	LoCoMo, MemoryAgentBench
MemoryOS	GitHub Website	✔	✔	✘	hierarchical (S/M/LTM)	LoCoMo, MemoryBank
MemOS	GitHub Website	✔	✔	✘	tree memory + memcube	LoCoMo, PreFEval, LongMemEval, PersonaMem
Zep	GitHub Website	✔	✔	✘	temporal knowledge graph	LongMemEval
LangMem	GitHub Website	✔	✔	✘	core API + manager	-
SuperMemory	GitHub Website	✔	✔	✔	vector + semantic	-
Cognee	GitHub Website	✔	✔	✔	knowledge graph	-
Memary	GitHub Website	✔	✔	✘	stream + entity store	-
Pinecone	GitHub Website	✔	✘	✘	vector database	-
Chroma	GitHub Website	✔	✘	✔	vector database	-
Weaviate	GitHub Website	✔	✘	✔	vector + graph	-
Second Me	GitHub Website	✔	✘	✘	agent ego	-
MemU	GitHub Website	✔	✔	✔	hierachical layers	-
MemEngine	GitHub	✔	✔	✔	modular space	-
Memori	GitHub Website	✔	✔	✘	memory database	-
ReMe	GitHub Website	✔	✔	✘	memory management	-
AgentMemory	GitHub Website	✔	✔	✘	memory management	-
MineContext	GitHub Website	✔	✔	✔	context engineering	-
Acontext	GitHub	✔	✔	✔	context engineering + skill learning	-
PowerMem	GitHub	✔	✘	✔	oceanbase	-
ReMe	GitHub	✔	✔	✘	agentscope	BFCL, AppWorld
HindSight	GitHub	✔	✔	✘	parallel retrieval + reflection	-

Table 9 Overview of representative open-source memory frameworks for LLM-based agents. The table compares widely used frameworks in terms of the types of memory they support (factual vs. experiential), multimodality, internal memory structure, and reported evaluation benchmarks. **Fac.** and **Exp.** denote factual and experiential memory, respectively, **MM.** indicates multimodal memory support, and **Structure** summarizes the core memory abstraction or organization mechanism adopted by each framework. **Evaluation** lists publicly reported benchmarks used to assess memory-related capabilities, when available.

Positions and Frontiers

- A shift **from memory retrieval to memory generation**

Instead of only retrieving stored memory, future agents may actively generate, compress, and reorganize memory to better support the current task.

- **Automated memory management**

Current systems are still largely hand-crafted. A key next step is to let agents automatically decide what to store, update, consolidate, and forget.

- **Reinforcement learning for memory control**

The paper highlights a shift from heuristic memory pipelines to RL-driven memory policies, where agents learn when and how to manage memory through interaction.

- **Multimodal and shared memory**

As agents move into richer environments, memory should support text, image, video and other modalities, and in multi-agent settings, agents may need shared memory spaces for coordination and collaboration.

- **Memory for world modeling**

Memory should not only store past facts or experiences, but also help agents build a more coherent world model of the environment, enabling better long-horizon reasoning and planning.

- **Trustworthy and human-inspired memory**

Future work must address memory errors, hallucinated memories, privacy and reliability, while also drawing inspiration from human cognition, such as consolidation and more structured long-term memory organization.

Zero-RAG: Towards RAG with **Zero Redundant Knowledge**

Authors: Qi Luo, Xiaonan Li, Junqi Dai, Shuang Cheng, Yining Zheng, Xipeng Qiu

Presented By: Jacob Huynh

Background Context

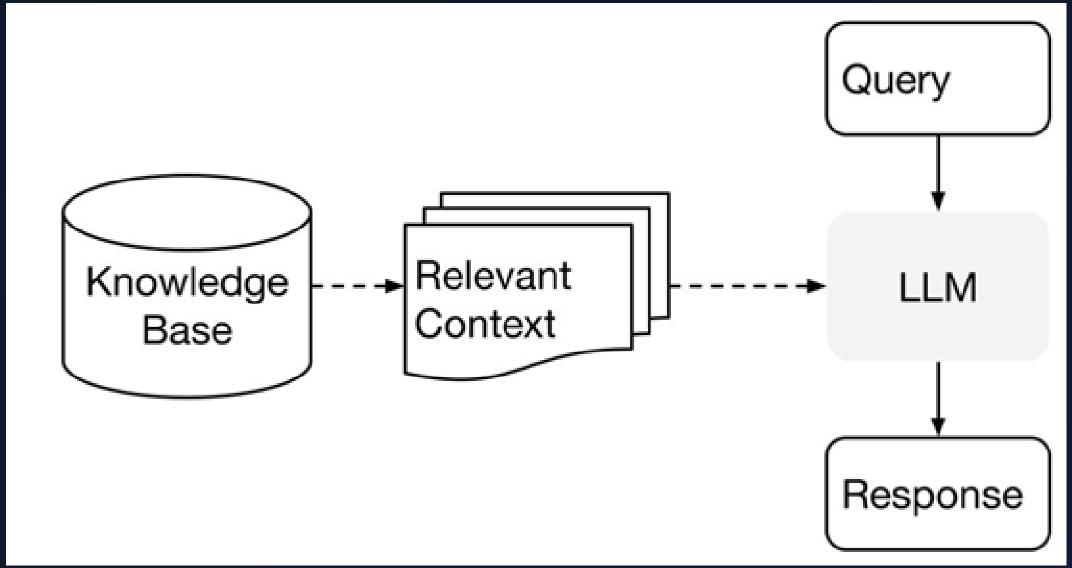
Understanding the foundation: Retrieval-Augmented Generation

What is RAG?

Retrieval-Augmented Generation (RAG) enhances Large Language Models (LLMs) by grounding them in external knowledge, bridging the gap between frozen internal weights and dynamic data.

- **1. Retrieve:** User queries are searched against an external database (via vector embeddings) to find highly relevant documents.
- **2. Augment:** These documents are injected directly into the LLM's prompt context.
- **3. Generate:** The LLM synthesizes a final answer using its internal reasoning and the factual context.

The Goal: Reduce hallucinations and provide accurate, domain-specific answers.



The Redundancy Problem

Why more data isn't always better for modern LLMs.

Rapid Scaling of LLM Knowledge

100

Days to Double

The Density Problem

As LLMs are trained on more data, the amount of world knowledge stored inside their parameters keeps growing. Recent research suggests this internal knowledge is doubling roughly every 100 days.

What used to be useful external context is increasingly just a repeat of what the model already has.

The Redundancy Problem

Massive Overlap

- As LLMs scale, their internal memory grows, reducing the need for external data sources.
    - Llama 3.3-70B is able achieve at least 40% accuracy on Wikipedia-related questions.
- Indexing and retrieving information the model already knows is computationally expensive, slow, and redundant.

The Performance Drop

- Surprisingly, retrieving and feeding the LLM information it *already knows* actively distracts the model.
- The researchers observed that performance on "mastered" questions drops by **~20 percentage points** when redundant knowledge is forcefully added to the LLM's context window.

Formal Task Definition

1. Isolating Redundant Data

Redundancy occurs when external passages overlap significantly with LLM internal knowledge K_M.

$$\mathcal{D}_{\text{redundant}} = \{d_i \in \mathcal{D} \mid \text{Overlap}(d_i, \mathcal{K}_{\mathcal{M}}) \leq \tau\}$$

2. Retained Database

Our objective is to prune redundant knowledge to produce a concise, optimized corpus.

$$\mathcal{D}_{\text{retained}} = \mathcal{D} \setminus \mathcal{D}_{\text{redundant}}$$

Subtracting redundant documents leaves only the information the model actually needs.

Zero-RAG Architecture

A three-part solution to streamline retrieval.

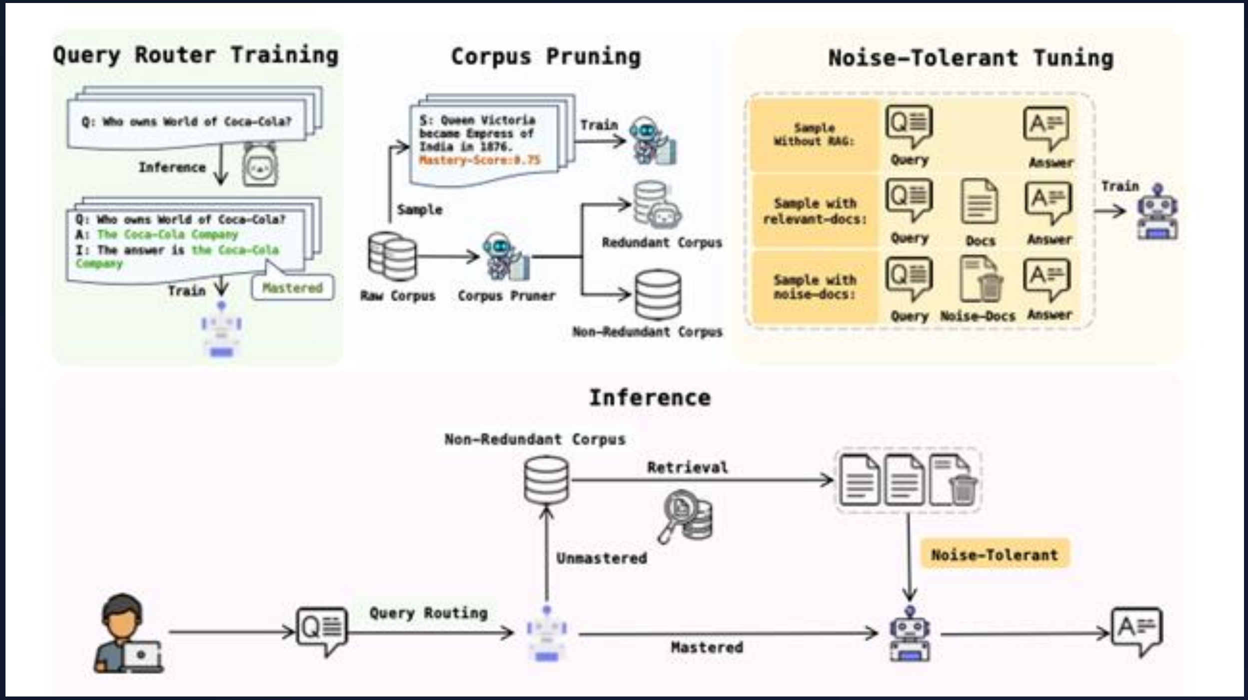
System Overview

A Three-Part Pipeline

1. Mastery-Score (Corpus Pruning): Evaluates and physically removes passages from the database that the LLM already knows.

2. Query Router: A gatekeeper that determines if an incoming query can be answered purely from internal memory, bypassing retrieval entirely.

3. Noise-Tolerant Tuning: Fine-tunes the LLM to remain robust and trust its internal knowledge even if irrelevant or distracting documents are retrieved.



Module 1: The Mastery-Score

How do we systematically test what the LLM actually knows? Memorization does not equal utilization.

1. Construct QA Pairs

Take a sentence from the database. Use an LLM (e.g., GPT-4o-mini) to generate multiple Question-Answer pairs directly based on that sentence.

2. Compute EM Score

Prompt the main LLM with the generated questions. Calculate the Exact Match (EM) score to see if it provides the correct short answer.

3. Train Proxy Model

Testing 138M Wikipedia sentences is too costly. Train a smaller 7B regression model to predict this score based purely on reading the sentence text.

Creating QA Pairs and EM Score Calculation

QA Pairs

"Queen Victoria became Empress of India in 1876."

Who became Empress of India in 1876? -> Queen Victoria

In what year did Queen Victoria become Empress of India? -> 1876

...

Field	Content
Wiki Sentence	*"Queen Victoria became Empress of India in 1876."*
Q&A Pairs	**Q1:** Who became Empress of India in 1876? **A1:** Queen Victoria **Q2:** In what year did Queen Victoria become Empress of India? **A2:** 1876 **Q3:** What title did Queen Victoria acquire in 1876? **A3:** Empress of India **Q4:** Which British monarch became Empress of India in 1876? **A4:** Queen Victoria
Predictions	**P1:** The answer is Queen Victoria. She was proclaimed Empress of India in 1876. **P2:** Queen Victoria became the Empress of India in 1876. **P3:** The answer is: Empress of India. **P4:** The answer is Queen Victoria. She was proclaimed Empress of India by the Royal Titles Act 1876.
Eval Results	[Mastered, Mastered, Mastered, Mastered]
Mastery-Score	1.0

EM Score Calculation

When Llama 3-70B was asked four unique questions based on this sentence *without* any RAG context, it answered all 4 with **100% accuracy** -> Mastery Score of 1

$$M(s) = \frac{1}{n} \sum_{i=1}^{n} \mathrm{EM}(a_i, L(q_i)),$$

Training the Proxy Model

Bottleneck

Testing all 138 million Wikipedia sentences with the massive 70B model is extremely computationally expensive.

Solution

Researchers created a training subset mapped to the calculated EM scores and used it to train a smaller 7B parameter regression model.

- Ex: ("Queen Victoria became Empress of India in 1876.", 1)

Objective

The 7B model is fine-tuned to minimize Mean Squared Error (MSE) to learn to predict the larger model's redundancy score purely on the sentence text.

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \Big(f_\theta(s_i) - m_i \Big)^2,$$

Deploying the Proxy

Prediction

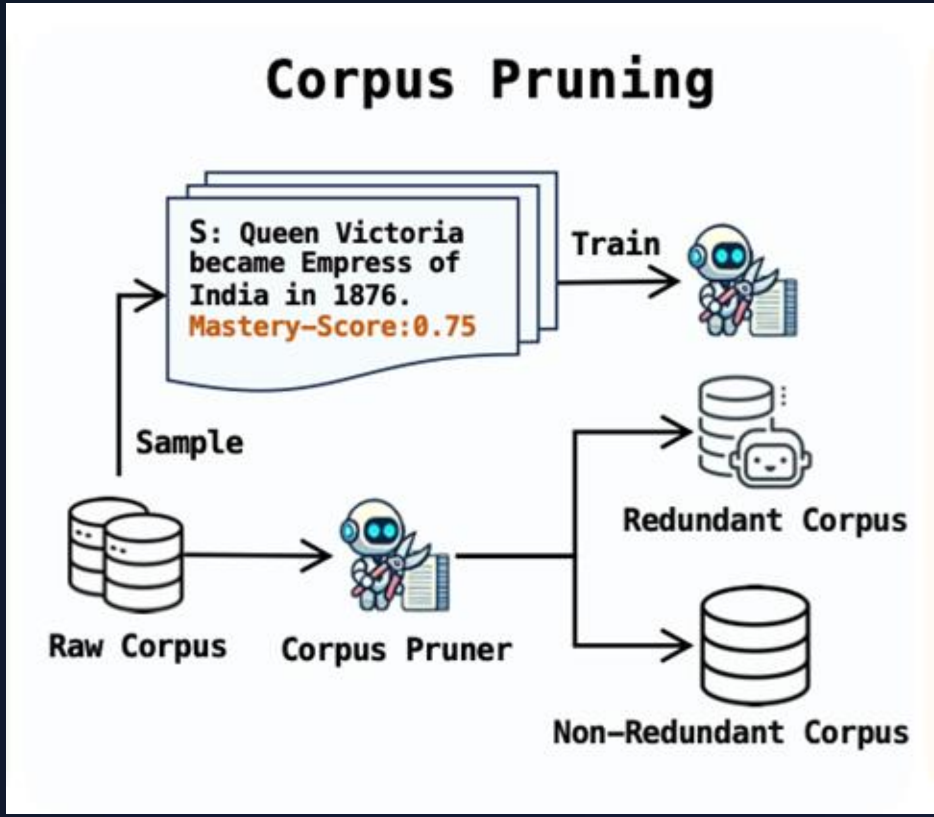
The trained 7B model can now take in raw database text and output a redundancy score from 0 to 1.

Dynamic Thresholding

Any sentence that receives a high score (above the pre-defined τ) is flagged as “mastered” by the LLM.

Result

These redundant sentences are permanently pruned, leaving a highly optimized and non-redundant corpus for retrieval

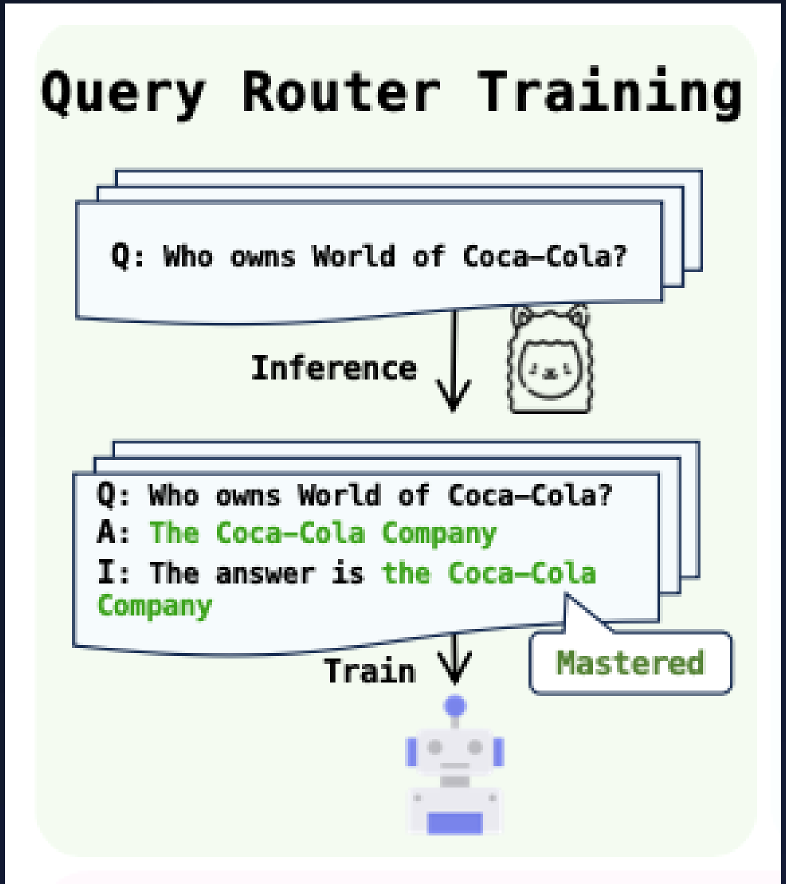


Module 2: The Query Router

The Intelligent Gatekeeper

Even with a pruned database, running a retrieval step for a known question wastes time and risks retrieving confusing noise. The Query Router bypasses retrieval entirely if the LLM is confident.

- **Training Data:** Queries are tested and labeled in binary: *Mastered* vs *Unmastered*.
- **Mechanism:** A neural network trained via binary classification loss.



Module 3: Noise-Tolerant Tuning

To bulletproof the LLM against bad search results, it is fine-tuned to ignore irrelevant documents and trust its internal memory using three scenarios.

1. Retrieval-Free

Input: Query q only.
Target: Answer a.

Forces the model to rely purely on its internal knowledge without any context.

2. Retrieval-Augment

Input: Query q + Relevant Docs r_p.
Target: Answer a.

Reinforces standard RAG behavior, extracting facts from good sources.

3. Noise-Suppression

Input: Query q + Irrelevant Docs r_n.
Target: Answer a.

Teaches the model to actively ignore unhelpful text and fall back on its memory.

Unified Fine-Tuning Loss

$$\mathcal{L} = \underbrace{-\mathbb{E}\big[\log p_\theta(a|q)\big]}_{\text{Retrieval-Free}} \underbrace{-\,\mathbb{E}\big[\log p_\theta(a|q, r_p)\big]}_{\text{Retrieval-Augment}} \underbrace{-\,\mathbb{E}\big[\log p_\theta(a|q, r_n)\big]}_{\text{Noise-Suppression}}$$

Each term addresses a distinct failure mode:

- Without the retrieval-free term, the model becomes dependent on always having context.
- Without the retrieval-augment term, it forgets how to use good documents.
- Without the noise-suppression term, it gets confused by irrelevant ones.

Zero-RAG Inference Pipeline

1. Pre-Processing

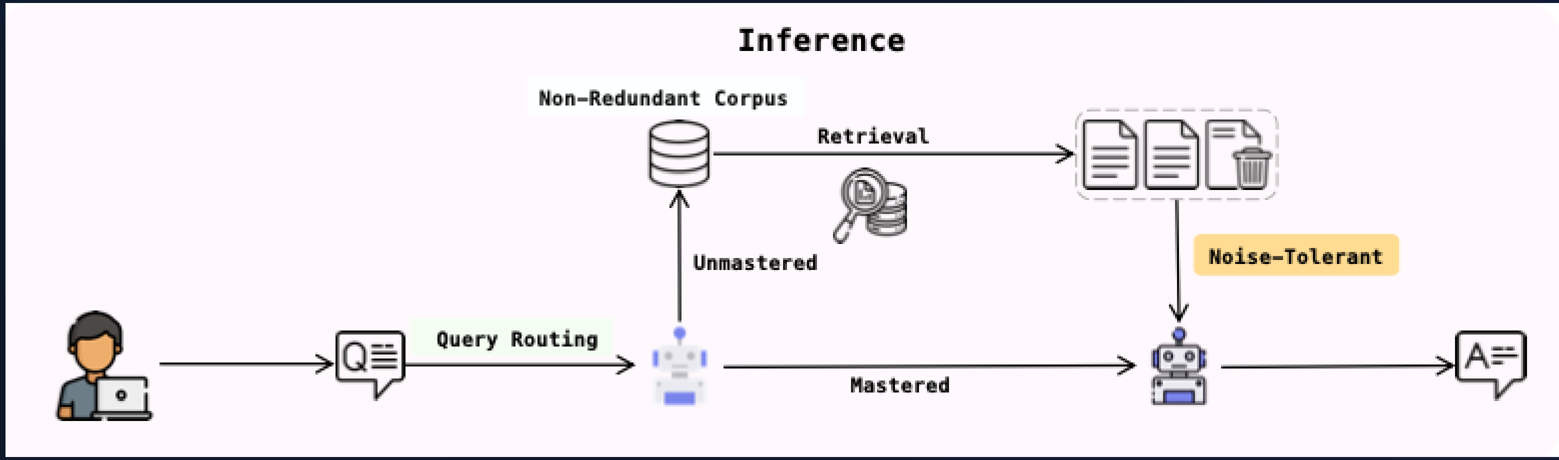
The original Wikipedia database is processed. The Corpus Pruner assigns Mastery-Scores and deletes high-scoring sentences permanently.

2. Query Routing

A user submits a query. The Query Router evaluates it in real-time to check if the LLM already possesses the internal knowledge to answer it.

3. Final Inference

If "Mastered", the LLM answers from memory. If "Unmastered", it searches the pruned DB and uses its noise-tolerant training to answer safely.



Experiments & Results

Proving the efficiency and accuracy of Zero-RAG.

Experimental Settings

Four Distinct Datasets

- **EntityQuestions:** Simple, structured queries about specific entities.
- **TriviaQA:** Large-scale trivia questions requiring broad world knowledge.
- **PopQA:** Questions generated from templates based on popular entity pairs.
- **HotpotQA:** Complex queries needing multi-hop reasoning across documents.

Implementation Details

- **Database:** Entirety of Wikipedia, segmented into exactly 138,390,600 sentences.
- **QA Generation:** GPT-4o-mini used to generate training pairs.
- **Models Tested:** Llama3-70B, Llama3.3-70B, and Llama3-8B.
- **Retriever:** stella_en_400M_v5 (fetching top 20 candidates).

Main Results (Llama 3.3-70B)

20920Method	PopQA	HotpotQA	TriviaQA	EntityQuestions
		Llama3-70B		
Llama3-70B-Instruct	14.08	43.71	80.66	51.91
+ Retrieval	15.62	41.40	76.44	48.25
Noise-Tolerant Tuning	25.21	42.70	81.43	54.14
+ Retrieval	39.36	49.67	81.90	65.25
Zero-RAG (No Pruning)	31.72	45.00	81.80	60.82
Zero-RAG (- 10% Corpus)	30.81	43.84	81.50	58.92
Zero-RAG (- 30% Corpus)	30.67	43.30	81.00	57.82
Zero-RAG (- 50% Corpus)	30.32	41.93	80.53	56.11
Zero-RAG (- 70% Corpus)	29.48	40.69	80.40	55.41
		Llama3.3-70B		
Llama3.3-70B-Instruct	16.25	46.20	81.43	52.01
+ Retrieval	16.35	43.35	79.32	50.71
Noise-Tolerant Tuning	32.77	47.50	82.19	55.38
+ Retrieval	38.94	49.12	81.50	65.16
Zero-RAG (No Pruning)	35.78	51.28	82.69	63.70
Zero-RAG (- 10% Corpus)	35.43	49.41	82.57	61.33
Zero-RAG (- 30% Corpus)	34.80	48.52	82.42	60.43
Zero-RAG (- 50% Corpus)	34.24	47.09	82.17	57.35
Zero-RAG (- 70% Corpus)	32.91	46.20	82.07	56.29

Key Findings

- Standard RAG actually *lowered* scores on HotpotQA, TriviaQA, and EntityQuestions due to the distraction penalty.
    - WIth Noise-Tolerant Tuning, RAG actually helps.
- Pruning **30%** of the database resulted in an average drop of less than 2 points compared to 0% pruned.
- On TriviaQA, deleting a massive **70%** of the database caused an insignificant point drop.

Ablation Studies

Method	TriviaQA		HotpotQA	
	Prune Ratio	EM	Prune Ratio	EM
Llama3.3-70B-Instruct	-	81.43	-	46.20
+ Retrieval	0%	79.32	0%	43.35
Zero-RAG	30%	$\boxed{82.42}$	30%	$\boxed{48.52}$
- Corpus Prune	0%	82.69	0%	51.28
- Query Router	30%	$\boxed{81.50}$	30%	$\boxed{43.35}$
- Noise-Tolerant Tuning	30%	$\boxed{80.55}$	30%	$\boxed{42.82}$

Key Findings

- **Removing Query Router:** Drops performance because the system performs unnecessary retrievals for known questions, pulling in noise.
- **Removing Noise-Tuning:** Causes the most severe drop. The model loses its robustness against irrelevant search results.

Retrieval Efficiency & Speed

Prune Ratio	HotpotQA	EntityQ	TriviaQA	Avg
0%	14.65	10.96	11.24	12.28
30%	10.89	9.09	9.00	9.66

time measured in seconds

Faster Backend Processing

- By pruning 30% of the redundant sentences from the massive Wikipedia database, the system experiences a marked decrease in latency across all datasets.

Limitations

1. Sampling Bias in Proxy Training

The 7B classifier is only as good as the subset used to train it. If the sampled sentences over-represent common factual content and under-represent technical or niche domains, the classifier will systematically mislabel whole categories — potentially pruning knowledge the LLM actually needs.

2. Wikipedia-Only Evaluation

All experiments use Wikipedia as the corpus. It's unclear whether Zero-RAG generalizes to domain-specific corpora like medical literature, legal documents, or internal enterprise knowledge bases — where the LLM's internal knowledge overlap would likely be much lower, making aggressive pruning riskier.

3. Mastery-Score is LLM Specific

The pruned corpus is calibrated to a specific model (e.g. Llama3.3-70B). If you swap in a different LLM, the corpus needs to be re-pruned from scratch. This makes Zero-RAG expensive to maintain as models get updated, which happens frequently.

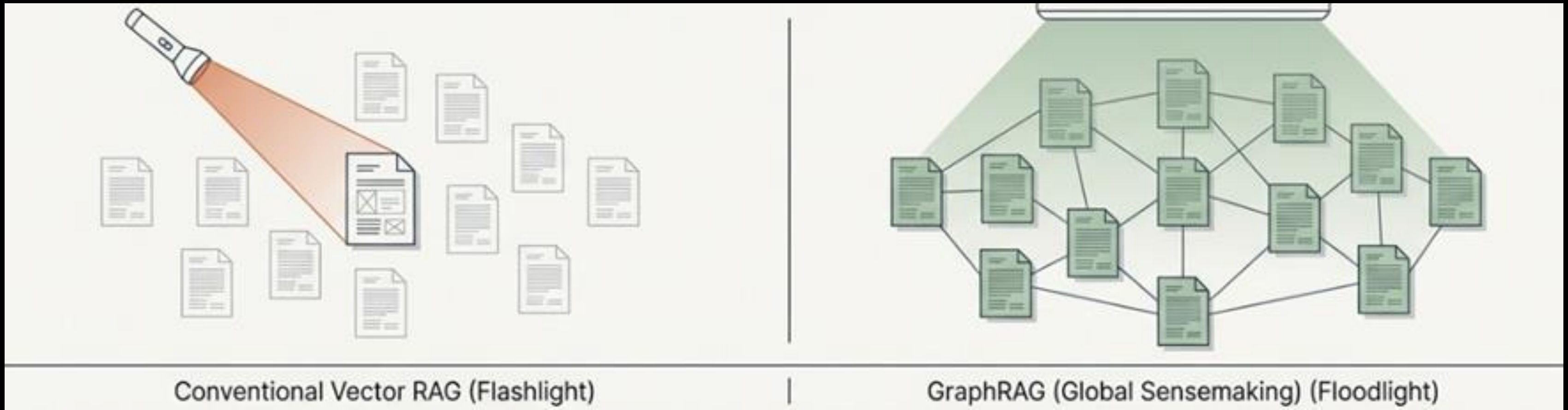
From Local to Global: A GraphRAG Approach to Query-Focused Summarization

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, Jonathan Larson

Presented by Henry Chen

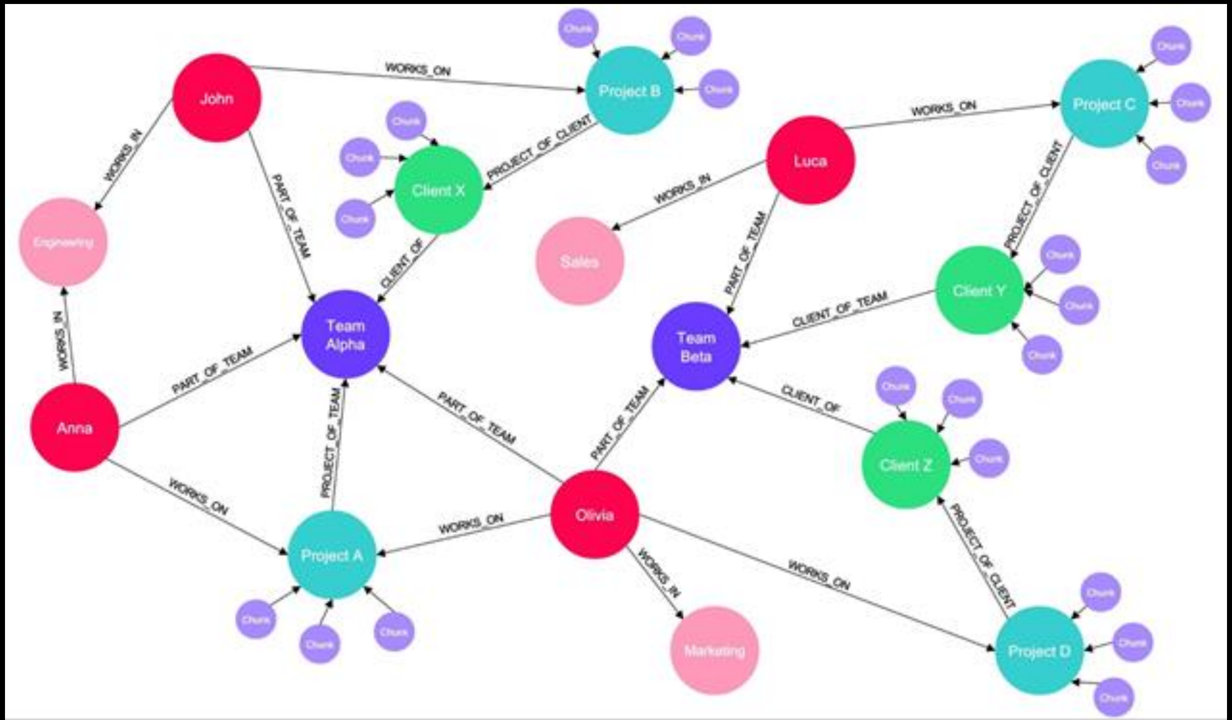
The Limits of Conventional RAG

- Conventional Vector RAG retrieves specific records based on semantic similarity.
- This works well for localizing specific facts, but fails at sensemaking queries that require a global understanding of the entire dataset.
- Sensemaking inherently requires reasoning over connections across the corpus to anticipate trajectories and synthesize themes



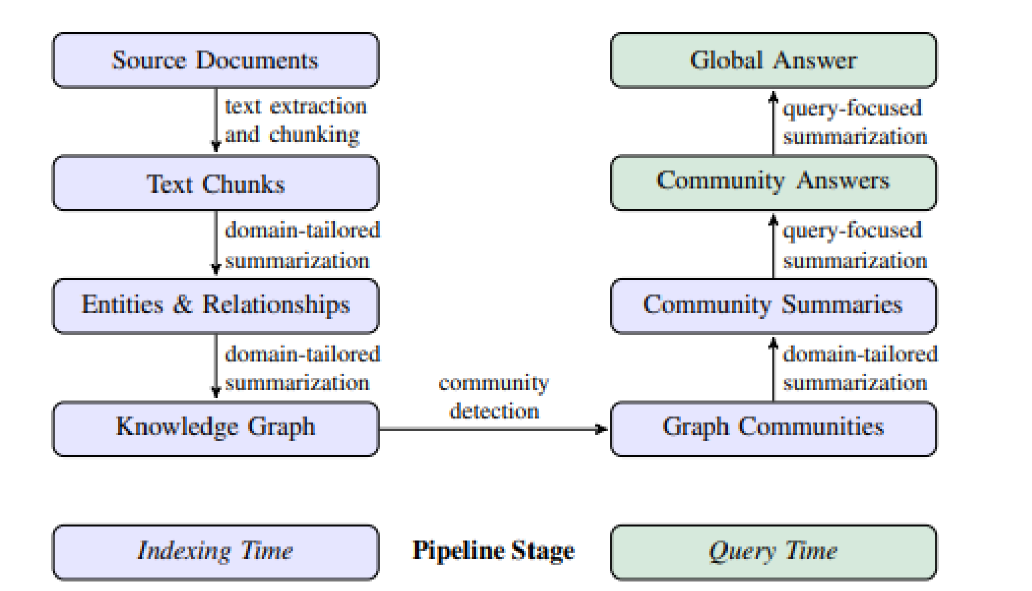
GraphRAG

- GraphRAG enables global sensemaking over large, private text corpora.
- It bridges the gap between retrieval and query-focused summarization.
- The system uses Large Language Models to construct a knowledge graph from the text, partition it into communities, and pre-generate summaries.



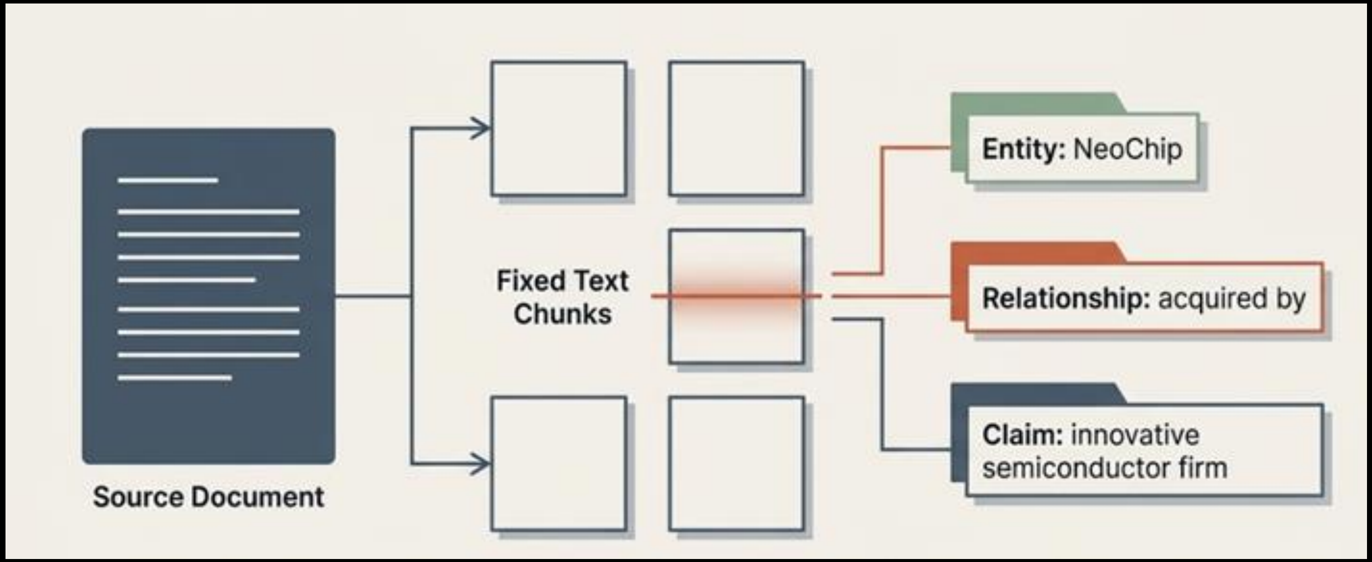
GraphRAG Pipeline Architecture

- The pipeline is divided into Indexing Time and Query Time.
- At indexing time, text is converted into a graph, partitioned into communities, and summarized.
- At query time, the system uses those pre-generated community summaries to construct a global answer.



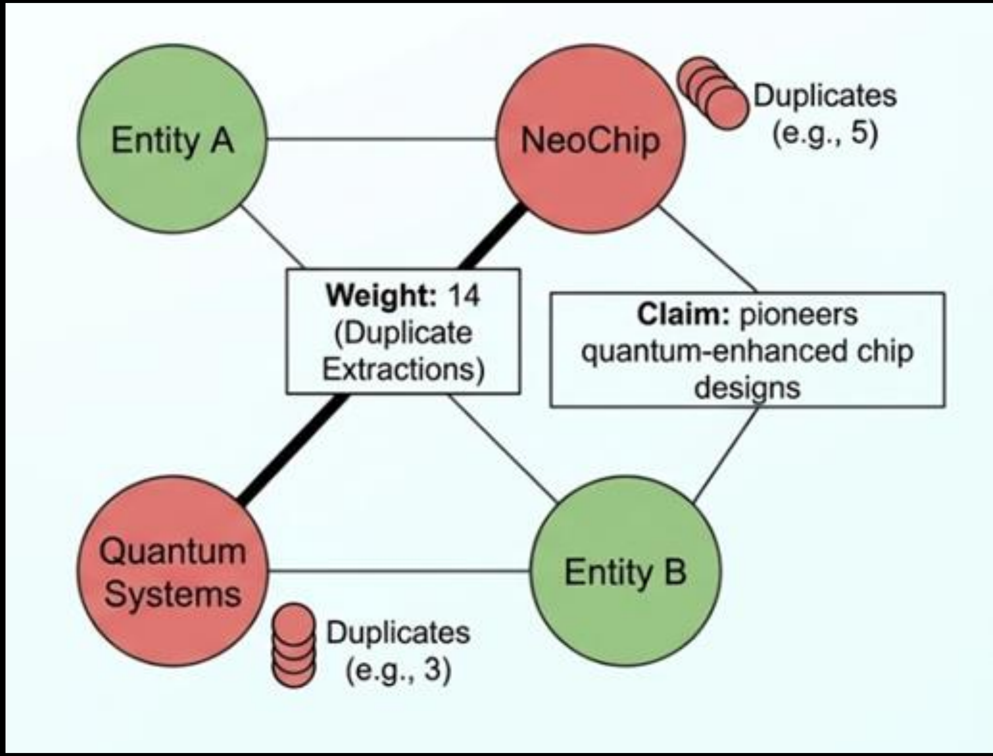
Text Chunking & Extraction

- Source documents are split into fixed text chunks.
- An LLM is prompted to extract instances of entities (people, places, organizations) and the relationships between them.
- The LLM also extracts "claims" which are important factual statements about the detected entities.



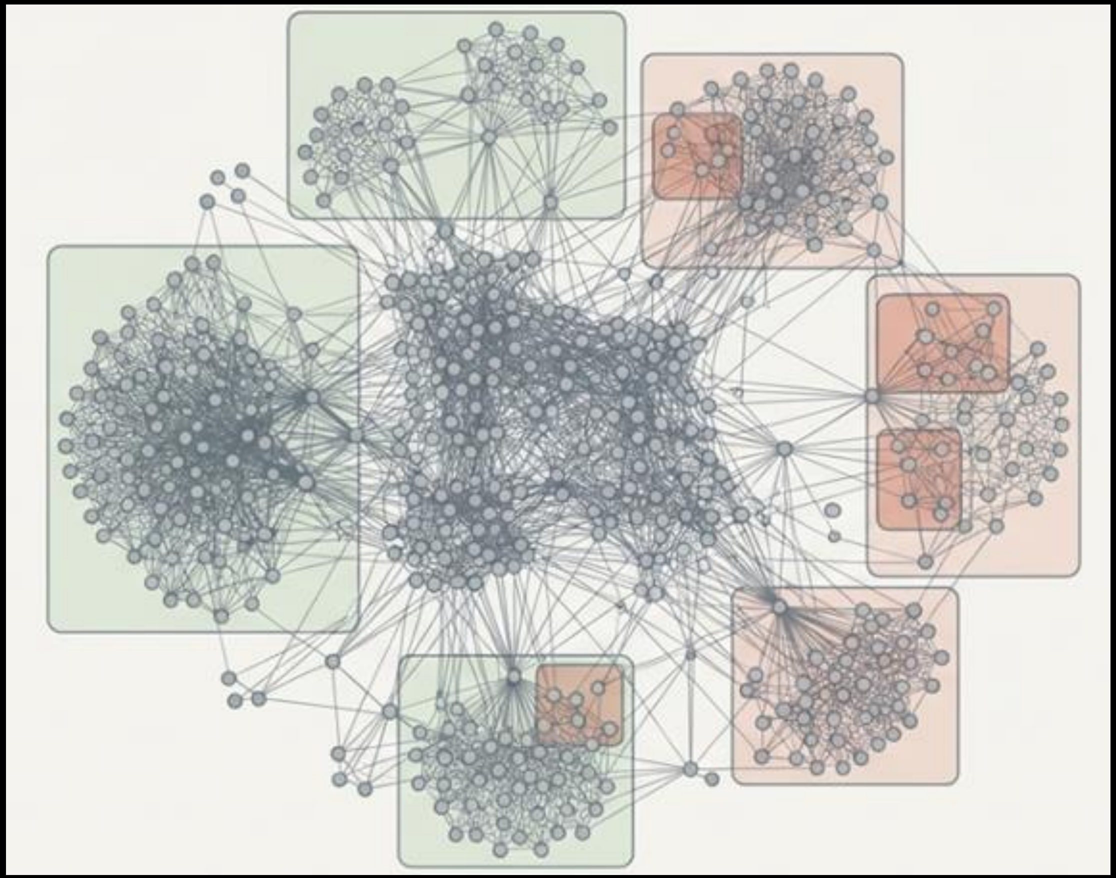
Building the Knowledge Graph

- Extracted element instances are aggregated into a single knowledge graph.
- Entity descriptions are summarized for each node.
- Relationships are aggregated into edges, where the number of duplicate relationship extractions acts as the edge weight.

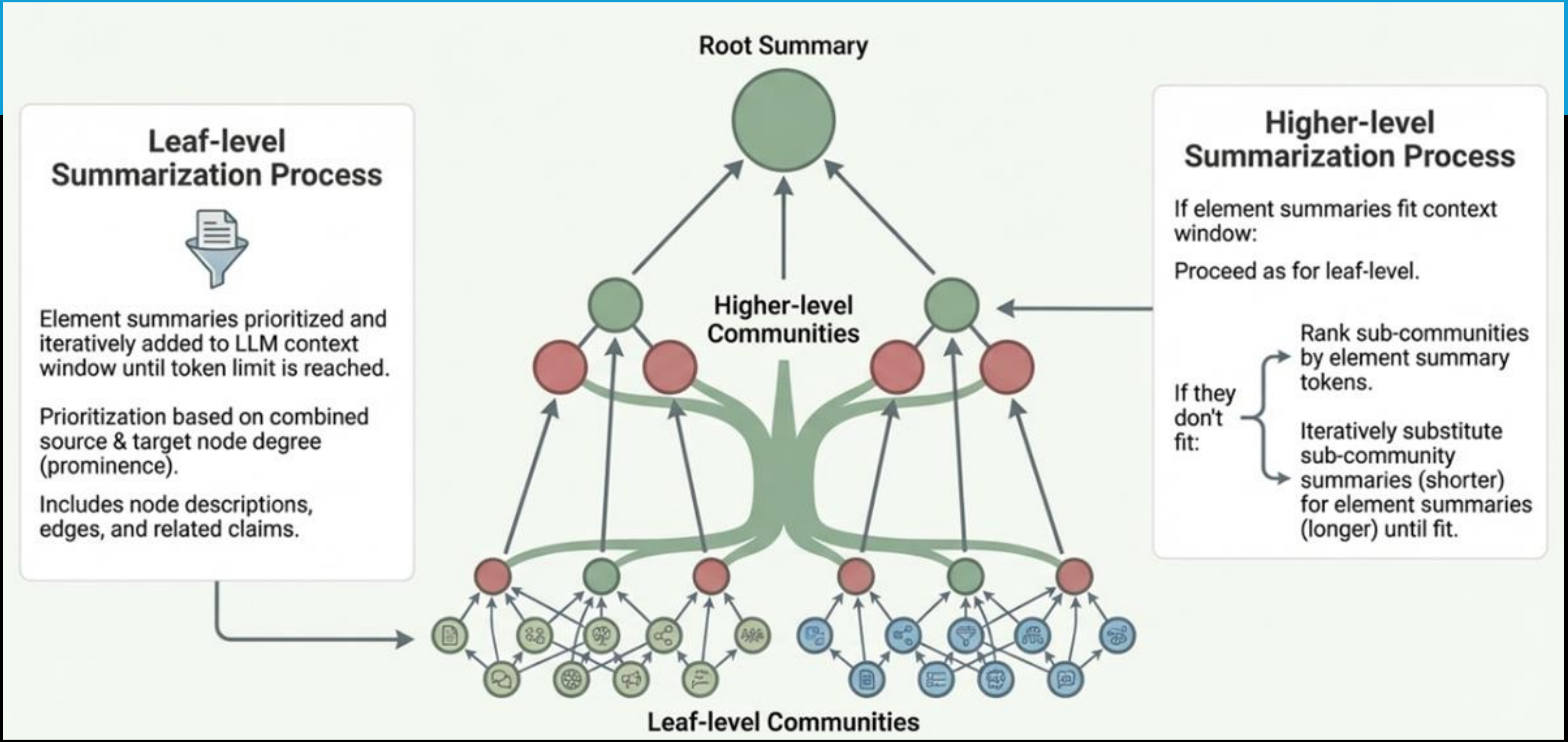


Community Detection

- The graph is partitioned using the Leiden community detection algorithm.
- This is done hierarchically, recursively breaking the graph down into sub-communities until reaching indivisible leaf communities.
- This nested modularity allows for "divide-and-conquer" global summarization

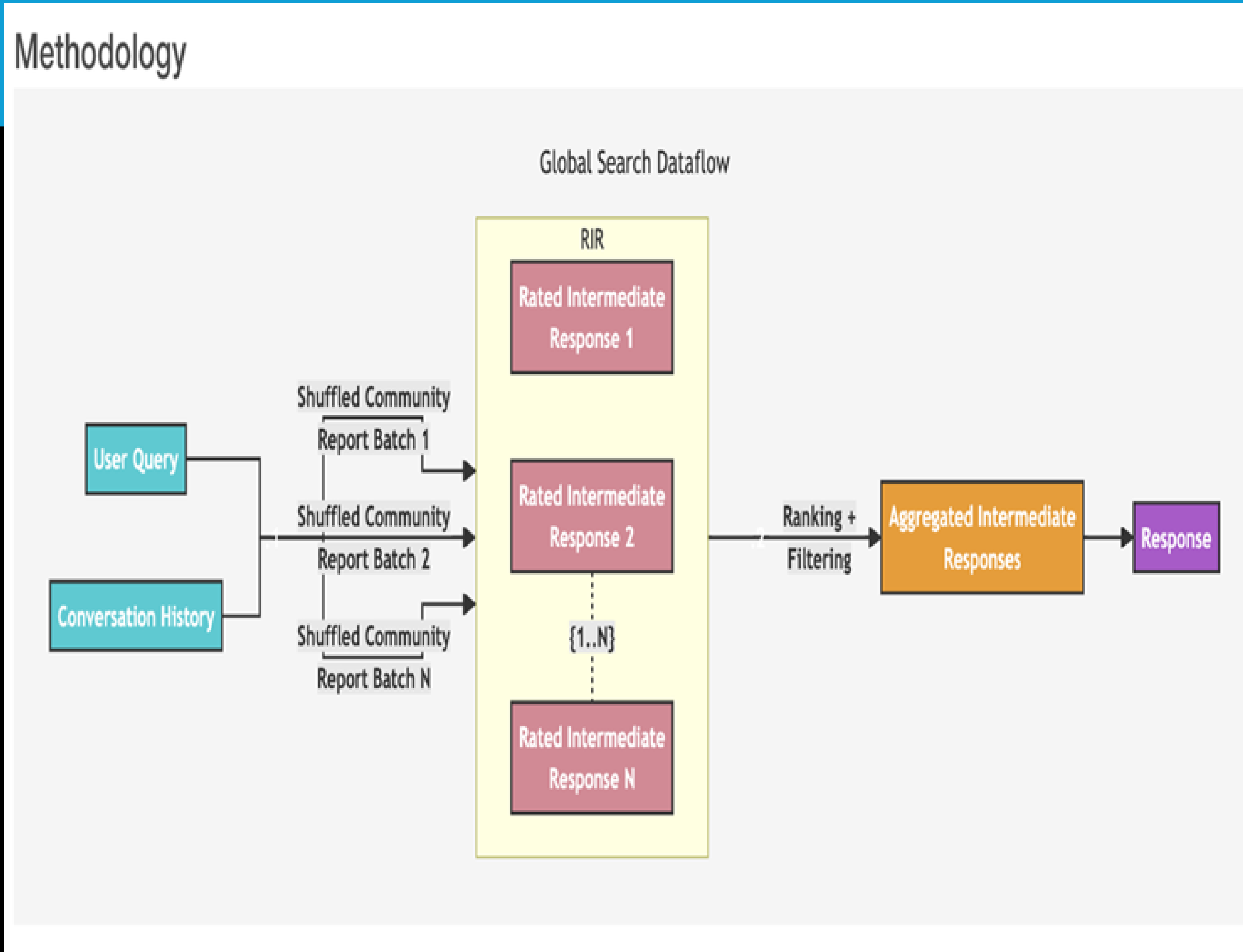


Community Summarization



Query Time

- Map Step: Community summaries are used in parallel to independently generate intermediate, partial answers to the query.
- Each partial answer is given a helpfulness score from 0-100 by the LLM.
- Reduce Step: The most helpful intermediate answers are sorted and combined into a final context window to generate the global answer.



The Evaluation Challenge: From Status Quo to Innovation

The Status Quo

Standard QA Benchmarks
(e.g., HotPotQA, MultiHop-RAG)

- Built for explicit fact retrieval (Vector RAG).
- Focuses on finding specific, localized answers.

The Missing Link

The Sensemaking Problem

- No "gold standard" or ground-truth available.
- Open-ended, thematic questions are subjective and lack a single right answer.

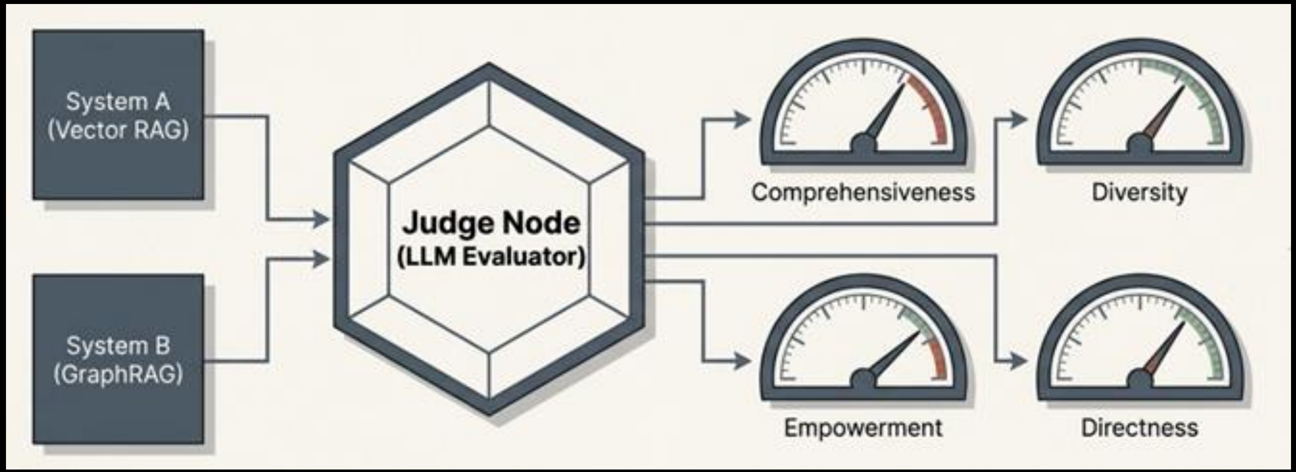
The Required Innovation

Adaptive Benchmarking

- Need a method to dynamically generate test queries.
- Queries must be tailored specifically to the target corpora.

Adaptive Benchmarking & LLM as a Judge

- The LLM infers potential system users and their tasks to generate 125 corpus-specific sensemaking queries per dataset.
- Performance is evaluated using an "LLM-as-a-judge" approach to compare generated answers from competing systems head-to-head.



Evaluation Criteria

Comprehensiveness

How much detail is provided to cover all aspects of the question.

Diversity

How varied and rich the answer is in providing different perspectives.

Empowerment

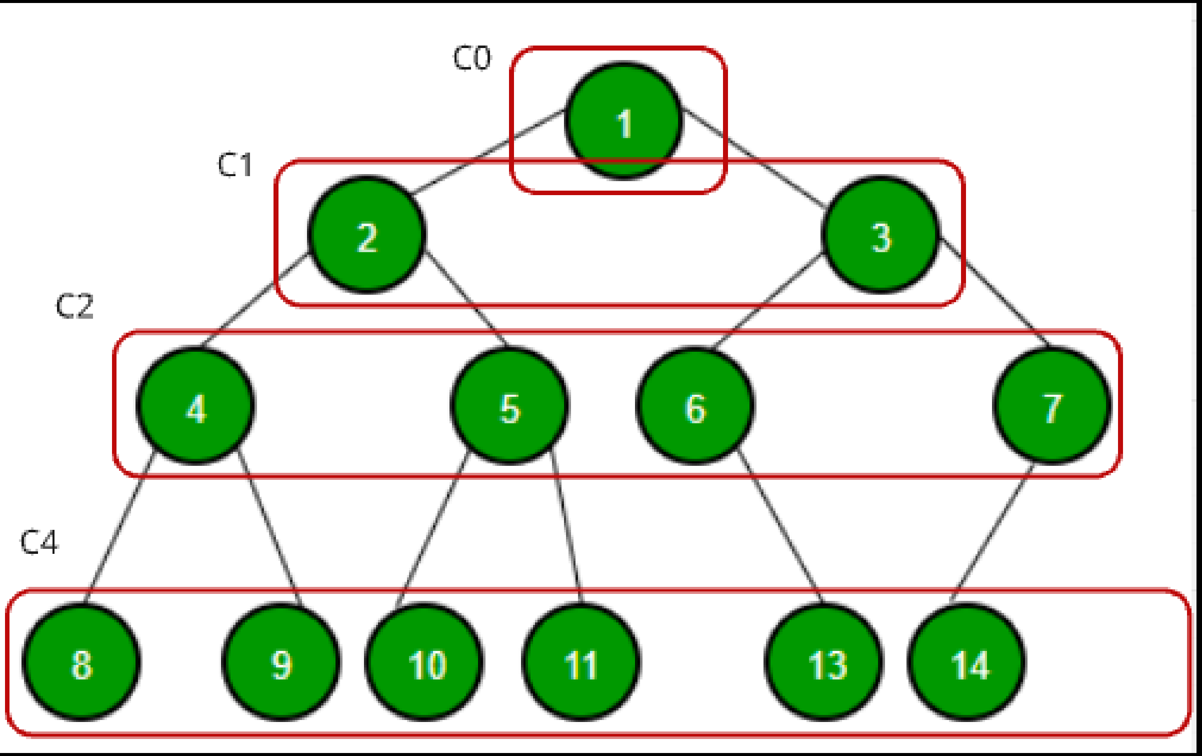
How well the answer helps the reader make informed judgments.

Directness (Control)

How specifically and concisely the answer addresses the question (used as a control reference against comprehensiveness).

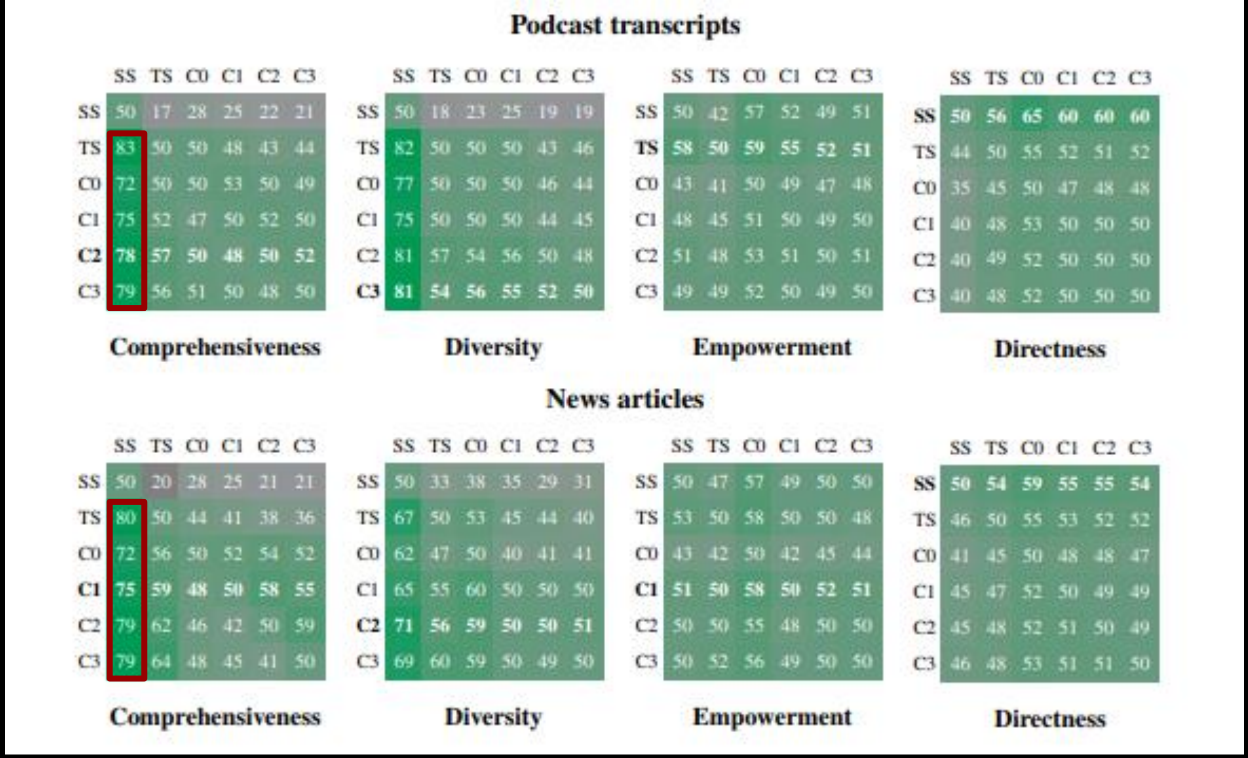
Experimental Setup

- Datasets: Two real-world datasets in the 1 million token range: Podcast transcripts and a News article benchmark.
- Baselines: GraphRAG was tested at four community levels (C0 to C3) against a naive Semantic Search (SS/vector RAG) baseline and a direct Text Summarization (TS) baseline.
- All queries used a fixed 8k context window to ensure fair comparisons.



Head-to-Head Win Rates

- GraphRAG global approaches significantly outperformed conventional vector RAG on comprehensiveness and diversity.
- Comprehensiveness win rates over vector RAG ranged from 72-83% for Podcasts and 72-80% for News.



Context Window Efficiency

- Map-reduce summarization on raw source text (TS) is the most resource-intensive approach.
- Root-level community summaries (C0) require dramatically fewer tokens per query (up to 97% fewer).
- GraphRAG provides highly efficient iterative query answering with a very modest drop in performance compared to raw text summarization.

	Podcast Transcripts					News Articles				
	C0	C1	C2	C3	TS	C0	C1	C2	C3	TS
Units	34	367	969	1310	1669	55	555	1797	2142	3197
Tokens	26657	225756	565720	746100	1014611	39770	352641	980898	1140266	1707694
% Max	2.6	22.2	55.8	73.5	100	2.3	20.7	57.4	66.8	100

Verifying Subjective Results

- Comprehensiveness: Measured by the average number of factual claims extracted from the generated answers
- Diversity: Measured by grouping the extracted claims and calculating the average number of distinct claim clusters per answer.

Condition	Average Number of Claims	
	News Articles	Podcast Transcripts
C0	**34.18**	32.21
C1	32.50	32.20
C2	31.62	**32.46**
C3	33.14	32.28
TS	32.89	31.39
SS	25.23	26.50

Dataset	Distance Threshold	Average Number of Clusters					
		C0	C1	C2	C3	TS	SS
News Articles	0.5	**23.42**	21.85	21.90	22.13	21.80	17.92
	0.6	**21.65**	20.38	20.30	20.52	20.13	16.78
	0.7	**20.19**	19.06	19.03	19.13	18.62	15.80
	0.8	**18.86**	17.78	17.82	17.79	17.30	14.80
Podcast Transcripts	0.5	**23.16**	22.62	22.52	21.93	21.14	18.55
	0.6	**21.65**	21.33	21.21	20.62	19.70	17.39
	0.7	**20.41**	20.04	19.79	19.22	18.08	16.28
	0.8	**19.26**	18.77	18.46	17.89	16.66	15.07

Conclusion & Future Work

- Combining knowledge graph generation with summarization effectively scales to support human sensemaking over entire corpora.
- Future iterations could involve hybrid RAG schemes that combine embedding-based local matching with GraphRAG's community reports.
- Exploratory "drill-down" mechanisms could be implemented to allow users to follow the information scent from higher-level to lower-level summaries.

Questions?