BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Lifelong Memory Architectures for Large Language Model Based Generative AI Systems: A Survey and Comparative Analysis

[1]Mayur Pawar, [2]Prof. Poonam Pawar

[1]Student, [2]Mentor
[1]Email: mayurpawar9850@gmail.com
[2]Email: poonampawar@dypic.in
[1]Dept. of Artificial Intelligence and Data Science,
[1]Ajeenkya DY Patil School of Engineering, Pune, Maharashtra, India

Abstract

Large Language Models (LLMs) form the basis of multiple generative applications today, but these systems are mostly stateless which means that information keeps outside the context window disappears when the session is closed. Such limitations complicate personalization, make it difficult to maintain consistency over long interactions, and prevent the AI system from learning from its experience. Lifelong memory architectures address this problem by providing the LLM with an external, permanent memory that also includes means of information writing, retrieval, updating, consolidating, and forgetting without the need to retrain the model itself. This paper presents an overview of the developments in this area starting from early external-memory enhancement and reflective generative agents, and continuing with memory operating system, graph-shaped memory, and reinforcement learning of agent memory. We summarize our results, suggesting an operation-centered classification of such systems and compare such twelve representations with regard to scalability, retrieval efficiency, personalization, computational cost, and adaptability, discussing necessary technologies such as embeddings, approximate nearest neighbor index, knowledge graph, and retrieval-augmented generation.

Keywords : Generative AI, Large Language Models, Lifelong Learning, Long-Term Memory, Memory Architecture, Retrieval-Augmented Generation, LLM Agents

I. INTRODUCTION

Transformer based LLMs were designed to create texts, writing codes, and generate structured thinking. Nevertheless, even with those developments, LLMs are still similar to people who wake up every morning with no memory of what they did yesterday—they only know the information contained in the prompt. Specifically, there is nothing in the memories from the phase of pre-training that can be used within conversations—after the context is removed, everything disappears from the LLM's knowledge. Also, it is important to note that different segments of the prompt are not always used in the same way and with the same level of importance.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

The reality of its implications is apparent. The tutor assistant is not able to recall what the student already knows; the coding agent is repeating errors made in the previous week, and so forth. The tutor assistant is unable to remember what the student already understands; the coding agent is re-iterating mistakes made last week and so on. Just increasing the size of the context window won't solve the issue since attention cost goes up with the input size. length, long prompts are expensive to resend, and nothing is carried between independent sessions.

This sort of thing is referred to as "lifelong memory architecture," where memory is regarded as a separate and managed system that can hold useful pieces of information, preferences, and experience for a long period of time, being able to provide the model with them whenever it is needed. The description of the area has quickly gained popularity; between 2023 and 2026 the memory stream proposal [13], use of operating system-style paging [14], graph-based consolidation [9], [10], linked note networks [8] were put forward by researchers, identity-smart memory designs [5], and memory rules mechanisms learned by reinforcement learning [4]. The surveys have further organized these initiatives into several categories of learning [1]–[3], [7].

The contributions of this paper are as follows:

1) A coherent presentation about memory in LLM systems with a distinction between the types of memory, their functions, and policies for their usage (Section II).

2) A systematic review of some examples and their classification based on design philosophies, as well as the review matrix and comparative analysis (Sections III and VI).

3) A description of the technologies used, their applications, and an outline of unrealized research areas with discussion (Sections IV–XI).

The rest of the report is structured as follows. Section II presents background information and terminology while Section III reviews architecture. Section IV discusses supporting technologies whereas Section V highlights memory management techniques and design options. Section VI compares systems and gives an overview of evaluations while Section VII gives illustrative examples, and Section VIII provides real-life applications. Sections IX-XI look into privacy, challenges, and future work. Section XII gives the conclusion.

II. BACKGROUND AND TERMINOLOGY

A. The Limitations of the Context Window

The context window is akin to the memory of the machine learning system; it provides speedy access but is quite limited as contents disappear at the end of a session. Retrieval-augmented generation (RAG) improves upon this because it enables the model to reference an outside corpus during inference.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

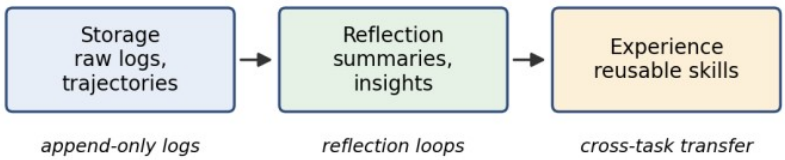


Fig. 1. Evolution of agent memory from storage to reflection to experience, following the progression described in [2].

Traditional RAG pipelines, on the other hand, access a relatively fixed set of documents. In comparison, a lifelong memory is written to while the memory system is in operation, thus making it necessary for the device to determine the contents worth saving, how to integrate it with pre-existing data, and when to remove it from memory.

B. A Taxonomy of Memory

Memory research in LLMs draws much from cognitive science. Many studies employ the traditional multi-store view of human memory, wherein short-term memory is viewed as a brief holding place and long-term memory a lasting system, and apply it to LLMs [7]. Memory in LLMs can be categorized in several ways.

- Regarding the distinction between parametric and non-parametric memory - parametric memory is present in the weights of models and can only change during training, whereas non-parametric memory exists outside in external memory sources (texts, vectors, and graphs) and can be modified at will.
- Extra distinction can be made between short-term and long-term memory - short-term memory (also known as working memory) is a dialogue already happening, while long-term memory refers to occurrences throughout sessions.
- Episodic memory, semantic memory, and procedural memory can also be differentiated - episodic memory makes a record of certain occurrences; semantic memory stores knowledge and preferences; while procedural memory holds skills or procedures.

Zhoubian *et al.* organize the field along representation, update dynamics, and persistence [1], while Luo *et al.* describe a trajectory from raw storage through reflection to reusable experience [2] (Fig. 1).

C. Core Memory Operations

All types of memory fulfill 6 functions despite their design:

1) Write/extract: identify the part of a conversational exchange to record in memory.
2) Index/organize: embed, label, or link the entries for future retrieval.
3) Retrieve: choose a few entries that are relevant for the current task.
4) Update/consolidate: update or combine information in the entries as new information is learned.
5) Forget: reduce the importance, archive, or delete information in old entries.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

6) Utilize: incorporate retrieved memory into the prompt and create the response.

The issue of retrieval as a mixture of recency, importance, and semantic significance is repeatedly mentioned [13]:

$$s(m, q) = \alpha_r \, \mathrm{rec}(m) + \alpha_i \, \mathrm{imp}(m) + \alpha_s \cos(e_m, e_q) \quad (1)$$

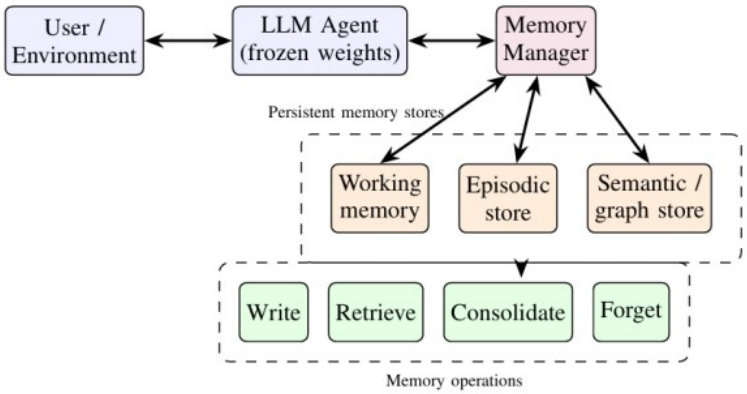


Fig. 2. Reference lifelong memory architecture for LLM-based systems. The memory manager mediates between the frozen model and stores operating at different time scales.

In the expression above e_m and e_q are the representations pertaining to the memory m and query q. Forgetting, in some cases, is represented in the form of the Ebbinghaus forgetting curve [12], in which the retention of the memory entry declines over time and is mitigated by the effect of a strengthening term S which increases with the number of retrievals of the memory entry.

$$R(t) = \exp(-t / S) \quad (2)$$

D. Scope and Survey Method

This research focuses on the memory aspect of LLM-based systems, which is related to the methods of storing, managing, and providing data to the model, not to development or retraining the foundation model itself. Studies were selected by two criteria: (i) they proposed or investigated long-term memory for LLMs or LLM agents, and (ii) they described the process of how information is created, pulled, updated, and lost; also, all studies should come from well-established conferences or known preprint repositories published in the period from 2023 to 2026 when LLM agent memory became an independent research area. Each of the studies has been summarized according to a given set of parameters such as methods, results, and limitations. The results of the investigations are displayed in the survey table (Table I). As the algorithms under review are tested on various types of tasks and models, we can provide qualitative not quantitative conclusions.

III. LIFELONG MEMORY ARCHITECTURES

A. A Unified Reference Architecture

Even though the systems presented differ in their specifics, most can be mapped to Fig. 2. An LLM agent maps the user's input and connects to a memory manager that has the write, retrieve, update, and forget

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

operations. Below the agent, there may be one or more stores with varying time scales. The model does not get retrained; only the stores and their management philosophy are modified.

B. External Memory Augmentation

LongMem retains the backbone LLM in its place and stores prior key-value representations in a permanent memory library; a simple external network learns to use cached states so that the system takes advantage of its memory far beyond its input constraints. While it improves reach, it offers little control over what to add and what to remove from memory.

C. Reflective Memory Streams

They save every observation made by Generative Agents as a natural language entry in a memory stream, recall memory entries using a scoring system similar to eq. (1), and periodically synthesize thoughts that they save back in the memory stream. The main merit of the family is that it is interpretable, while the main drawback is that searching an ever-growing memory stream can be costly.

D. Memory with Updating and Forgetting

MemoryBank keeps track of dialogue summaries and the ongoing modifications of a user's profile and a certain forgetting mechanism based on Ebbinghaus's curve reduce the amount of the less used entries in the database and strengthen the more frequently recalled entries [12]. Moreover, this was the first demonstration of forgetting functioning as a principle in itself rather than just being a limitation of the capacity of the storage device. Nevertheless, its significance and the rules of decay are designed manually.

E. Operating-System Style Hierarchies

MemGPT makes comparisons to how virtual memory functions in an operating system. The way that LLM operates is through use of various functions which enable transfer of information between two systems, where bounded "main context" is compared to RAM, while the unbounded external memory is like a disk [14]. MemoryOS simplifies the above into three classes classified according to how fast they are, i.e. short-term memory, mid-term memory, and long-term memory, and it offers "content promotion" from one tier to another based on "heat-like" metrics of memory that is being used [5]. Hierarchical scales prove to be quite successful although the introduction of additional scales requires additional understanding of many complexities.

F. Graph-Structured and Neuro-Inspired Memory

HippoRAG is motivated by the hippocampal indexing theory: it builds a knowledge graph from text and runs Personalized PageRank from query entities to surface connected evidence in a single retrieval step, which helps multi-hop questions [10]. Zep maintains a temporally aware knowledge graph in which facts carry validity intervals, making it better suited to evolving information [9]. Graph memories capture relations that flat vector stores miss, at the price of graph construction and maintenance cost.

G. Agentic and Self-Organizing Memory

The A-MEM makes it possible for an agent to manage its own memories according to the Zettelkasten approach, whereby every memory takes form of a note associated with either tags or keywords and links to other memory notes, and can even serve as a stimulus for modifying other memory notes [8]. The Mem0

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

system is regarded as more development-oriented since it extracts relevant data from dialogues and compares it against the entries already saved. Besides that, it has a graph component for relational information [6]. AgeMem goes even further in terms of memory handling: it utilizes memory actions in the structure of the agent's policies and enables it to conduct both short- and long-term memory attempts based on reinforcement learning [4].

H. Experience- Based Memory

Beyond facts, an agent may retain how it solved a task. Recent surveys identify the abstraction of raw trajectories into reusable, transferable experience as a central open problem [2], [3].

I. LITERATURE SURVEY MATRIX

Table I summarizes the core works examined in this study, listed from most recent to oldest.

Table I: Literature Survey Matrix of Representative Works

No.	Work / Year	Methodology	Key findings	Limitations
1	Zhoubian *et al.*, 2026 [1]	Architecture-based taxonomy (based on representation, update and persistence).	Unifies writing, retrieval, routing and consolidation in one view	Survey only; no common evaluation protocol
2	Zheng *et al.*, 2026 [3]	Lifelong-learning roadmap for LLM agents	Positions memory as essential for continual adaptation	Roadmap rather than a deployable architecture
3	Luo *et al.*, 2026 [2]	Three-stage view: Storage, Reflection, Experience	Follows memory from stored trajectories to distilled, reusable experience	Automatic abstraction and transfer between tasks are unsolved
4	Yu *et al.*, 2026 [4]	AgeMem: memory actions placed inside the agent's own policy	Agent decides when to store, fetch, revise, condense, or drop items	Reinforcement-learning training raises effort and cost
5	Wu *et al.*, 2025 [7]	Human-memory-inspired taxonomy	Maps human memory categories to LLM mechanisms	Taxonomy, not an implementable system

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

No.	Work / Year	Methodology	Key findings	Limitations
6	Xu *et al.*, 2025 [8]	A-MEM: linked, self-evolving notes	Dynamic organization of memories as new data arrives	Extra computation for linking and revision
7	Chhikara *et al.*, 2025 [6]	Mem0: extraction, consolidation, retrieval	Persistent memory for long-running agent dialogues	Privacy, conflict handling, evaluation still open
8	Kang *et al.*, 2025 [5]	MemoryOS: three tiers with dedicated modules	Short-, mid-, long-term tiers with storage, update, retrieval, generation	Added system complexity
9	Gutiérrez *et al.*, 2024 [10]	HippoRAG: knowledge graph + Personalized PageRank	Better long-term knowledge integration and multi-hop retrieval	Focuses on retrieval, not full memory lifecycle
10	Zhong *et al.*, 2024 [12]	MemoryBank with updating and forgetting	Supports long-term interaction with selective forgetting	Importance and decay rules are predefined
11	Park *et al.*, 2023 [13]	Generative Agents: memory stream + reflection	Stored experience and reflection improve long-horizon behavior	Simulated settings; scaling the stream is hard
12	Wang *et al.*, 2023 [15]	LongMem: frozen LLM, side network, cached memory bank	Uses history beyond the input window	Large-scale management and efficient retrieval

IV. ENABLING TECHNOLOGIES

Embeddings and semantic search. Text entries are mapped to dense vectors so that relevance becomes a similarity computation, as in Eq. (1). The quality of the embedding model directly bounds retrieval quality.

Vector indexes. The speed of exact nearest neighbor search decreases with an increase in memory size. Approximate methods such as graph-based indexing and GPU-accelerated search libraries sacrifice a little accuracy for a sublinear search time, and these methods are what vector-database memory usually relies on.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

RAG provides retrieval-driven generation. RAG is thought of as the means for memory retrieval when the information from the memory is plugged into the prompt. Lifelong memory can be regarded as an example of RAG where the memory mechanism is being updated by the agent in the course of its operation.

V. MEMORY MANAGEMENT POLICIES AND DESIGN TRADEOFFS

The way the memory works over a lifetime is governed more by the regulating rules than by the location of data. The present section discusses the four decisions that define the architectures referred to above.

A. Which Material to Store

Storing every piece of information is an easy task but is characterized by inefficiency and noise. Actual systems filter at the point of writing data. There are those that keep observations intact and apply summarizing methods afterwards while others may refer to and extract from which data is represented in the form of "prefers vegetarian food". Some may carry out a structured note with keywords and tags. While extraction reduces noise when storing and retrieving information it can also complicate the understanding of the point which was important.

B. How to Retrieve

There are several alternatives to the final value of Eq. (1), which can be defined as retrieval based on graph edges [10], retrieval from a time-based index [9] and retrieval based on tool invocation by the model [14]. One of the crucial engineering aspects is the number of retrieved entries k; if k is too small then evidence will not be gained, while if it is too high the model will get too many tokens and may be confused.

C. How to Update and Consolidate

In the case that some new data overlaps with a previously registered one, the entries can be added, updated, deleted, or neglected. This is formalized in Mem0 as a choice of add, update, delete, and no-op [6] while A-MEM allows replacing the metadata of the connected nodes with newly added notes [8]. However, the hierarchical structures can also aggregate brief memories into lengthy memory sections [5].

D. When and What to Forget

The process of remembering can lead to making the memory extensive, quick, and renewed. According to literature, time-dependent erosion with reinforcement upon reuse (Eq. (2)) [12], usage-dependent advancement and removal across layers [5], and elimination of facts by overruling information with specified periods of validity [9] are three mechanisms. Fig. 3 shows that the retention force constant S from Eq. (2) influences the lifespan of records - an infrequently used entry (small S) dies out fast. measurements, but rather conceptual, based on the equation.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

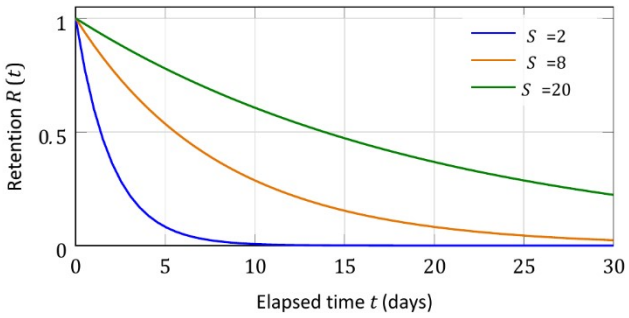


Fig. 3. Conceptual retention curves for three different strength values as per Eq. (2). As the entries are reused, S gets larger, making the curve flatter.

E. A Common Memory Update Procedure

In Algorithm 1, the loop was described that is commonly used by many systems. In particular, the specific decisions: extraction, matching, forgetting are made by either heuristics or the LLM.

Algorithm 1 Generic Lifelong Memory Update Loop

Require: Interaction turn x, memory store M, threshold τ
$C \leftarrow$ EXTRACT(x) ▷ *candidate facts*
for each $c \in C$ **do**
  $N \leftarrow$ TOPK(M, c) ▷ *similar entries*
  if CONFLICTS(c, N) **then**
    UPDATE(M, c) ▷ *revise or invalidate*
  else if $\max_{m \in N} \cos(e_c, e_m) > \tau$ **then**
    MERGE(M, c, N)
  else
    INSERT(M, c)
  end if
end for
DECAY(M); ARCHIVE entries with $R(t)$ below limit
return M

F. Selection of Storage Representation

The storage representation affects all other decisions, as shown in the comparison of the three main representations in Table II.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Table II: Common Memory Representations

Type	Strengths	Weaknesses
Flat text / vector store	Simple, fast to build, scalable with approximate-search indexes	Weak on relations, time, and multi-hop questions
Knowledge graph	Explicit relations; supports multi-hop and temporal reasoning [9], [10]	Costly construction and maintenance; schema issues
Hierarchical tiers	Bounds search space; natural promotion and eviction [5], [14]	More components to tune; promotion rules are heuristic

VI. COMPARATIVE ANALYSIS

Table III gives a qualitative comparison of representative systems against the criteria set out in our objectives. The ratings are the authors' own reading of each design as described in the cited papers; they are not measured benchmark results, since the systems are evaluated on different datasets and protocols.

There are three things to be noted. To begin with, if explicit layers or integration exists (e.g. Mem0, MemoryOS, and Zep), there is likely to be better scaling than in the case of plain streams. Second point is that the way the graph is arranged makes it easier to find relevant information, yet creation of the graph brings additional costs. The last observation is that the systems displaying maximum flexibility (the adaptive systems A-MEM and AgeMem) are also the most expensive from the point of view of implementation thus proving the relationship of flexibility and simplicity.

Table III: Qualitative Comparison of Memory Architectures (High / Medium / Low)

System	Scal.	Retr. eff.	Personal.	Cost	Adapt.
LongMem [15]	M	M	L	M	L
Generative Agents [13]	L	L	M	H	M
MemGPT [14]	M	M	H	M	M
MemoryBank [12]	M	M	H	L	M
HippoRAG [10]	M	H	L	M	M
A-MEM [8]	M	M	H	H	H
Mem0 [6]	H	H	H	M	M
MemoryOS [5]	H	M	H	M	H
Zep [9]	H	H	H	M	H
AgeMem [4]	H	M	H	H	H

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Cost: High = more computation/storage overhead (less favourable).

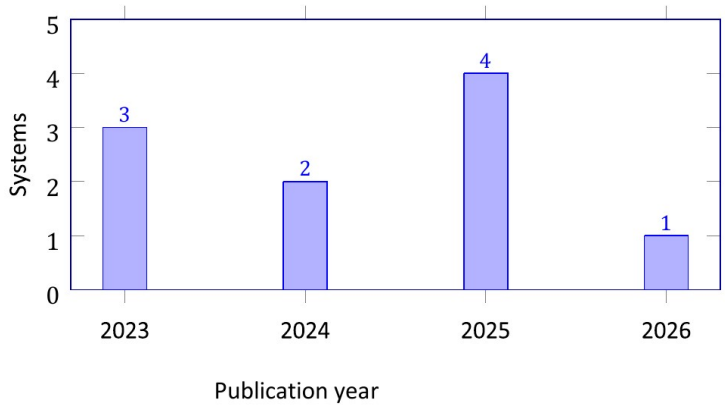


Fig. 4. The total number of memory systems that are represented in Table III, by the year of the original publication (2023: LongMem, Generative Agents, MemGPT; 2024: MemoryBank, HippoRAG; 2025: A-MEM, Mem0, MemoryOS, Zep; 2026: AgeMem).

A. Evaluation Benchmarks

The measurement of memory is a challenge. LoCoMo tests question answering and summarization capabilities in very long multi-turn dialogues [11], making it an appropriate testbed for memory evaluation. Despite the usefulness of these benchmarks, there are still few of them and they mainly concentrate on dialogues rather than the assessment of memory in agency and multi-modal environments.

B. Proposed Evaluation Dimensions

Considering that accuracy alone conceals crucial trade-offs, it is essential for memory evaluation processes to provide multiple measures at the same time, as shown in Table IV.

Giving information about costs and latencies in addition to accuracy is essential, as some of the most efficient approaches have both high accuracy and latency.

Table IV: Suggested Evaluation Dimensions for Lifelong Memory

Dimension	What it measures
Recall quality	Whether the needed entry appears among the top-*k* retrieved items
Answer accuracy	Correctness of the final response that uses retrieved memory
Temporal handling	Correct use of the latest fact when information changed over time
Abstention	Ability to say “unknown” when nothing relevant was stored
Latency and token cost	Added delay and prompt length per turn, including write-time LLM calls
Storage growth	Memory size as a function of interaction history
Forgetting quality	Removal of stale or redundant items without losing valuable ones
Privacy leakage	Rate at which sensitive stored data is exposed to unauthorized queries

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

However, it is necessary to utilize additional tasks of the model which enable data extraction, linking or summarization whenever required.

VII. ILLUSTRATIVE SCENARIO

To illustrate how the operations function, consider the example of a study-planning assistant that employs a multi-layer memory. The scenario is, of course, hypothetical and is only intended to illustrate the process of interaction among various stages of the process.

The first week sees the user providing a certain preference, which is recorded by the system as a fact. After about five weeks, another fact, which has no relation to the first one, is captured by the memory system. By the twelfth week this fact is said to have been entered in the memory system, it is reached during the process of updating that the two facts cannot coexist; thus, the new fact is said to have invalidated the previous one.

In the following week, when the user of the system demands to prepare a schedule, the current time preference and the known weak topic are found on retrieval, but the old morning preference is left unavailable. This is how a system that does not feature conflict resolution may end up bringing about conflicting decisions, thus illustrating why policies in the mentioned studies have been created.

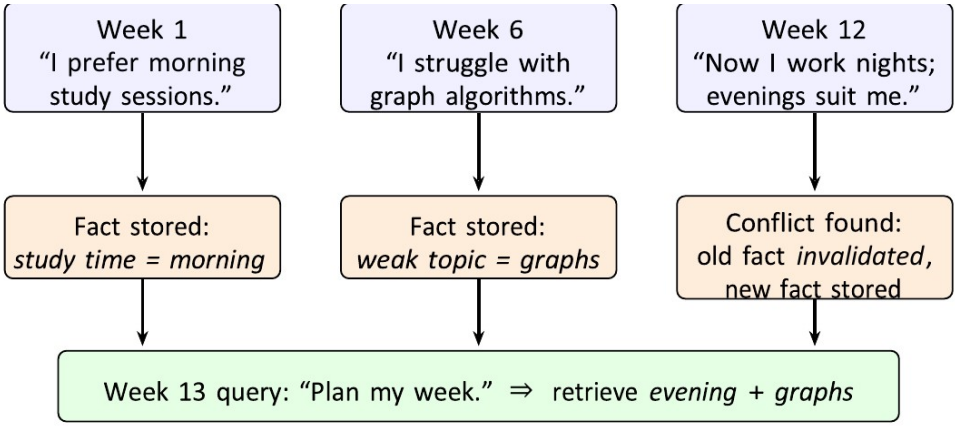


Fig. 5. Hypothetical walk-through: facts are extracted and stored, a later change triggers conflict resolution, and a new query retrieves only the current, relevant entries.

VIII. APPLICATIONS

Wherever there is an interaction that goes beyond a single instance, there is lifelong memory:

- Personal assistants recall habits, schedules, and preferences over months.
- Education model in which the instructor accompanies the student and views the learner struggling with the misunderstandings.
- Support service that maintains records of cases so that the customers won't need to keep explaining themselves.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

- Software development agents that have the memory of conventions and bugs that were fixed in the previous work.
- Agents that can apply past experiences to complex situations involving multiple steps and taking time to complete.
- Personal knowledge managing systems and long-term conversation partners.

IX. PRIVACY, SECURITY, AND ETHICAL CONSIDERATIONS

Persistent memory causes a conversational agent to become a long-standing holder of personal data, altering its risk profile. External memory is a queryable store of personal data in addition to whatever the model has learned in training. Table V shows the main concerns and their respective ways of avoidance.

In addition to the mechanical controls of the systems, the functions must enable the users to see and manage data. Those functions include the options to observe, change, and erase the stored data. Hence, it is important to design a forget process in a way that makes users able to check its effectiveness.

Table V: Privacy and Security Concerns in Persistent Memory

Concern	Description	Possible mitigation
Data leakage	Sensitive facts surface in responses to other users or contexts	Per-user isolation, access control, output filtering
Memory poisoning	Malicious or false content is written and later retrieved as trusted	Source tagging, validation before write, anomaly checks
Over-retention	Data kept longer than necessary or without consent	Retention limits, expiry, user-visible memory controls
Deletion requests	Users ask for removal, but copies persist in summaries or graphs	Provenance links, cascading deletion
Inference risk	Combined stored facts reveal sensitive attributes	Minimization, redaction, encryption at rest

X. CHALLENGES AND RESEARCH GAPS

Based on the reviewed literature, we identify the following gaps.

1) **Lack of a standard memory architecture.** Each system has its own memory architecture, memory model, and interface, making comparison and reuse difficult [1].
2) **Large scale retrieval.** When scale increases, search performance drops, and relevance may not be the most effective metric.
3) **Principle of forgetting.** The principle of forgetting is still in the open, and many decay and deletion rules are determined manually [12].
4) **Balancing short- and long-term.** It remains largely heuristic to choose what should be preserved in the prompt, what needs to be preserved, and how to proceed [4], [5].
5) **Regular updates and consolidation** while keeping lessons learned in memory.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

6) **Cost of computation and storage;** especially when applied to large language models for extraction, linking, and summarization at every turn [8].

7) **Conflicting and outdated memories.** The processes of reasoning about time and origin may change as preferences change or facts become obsolete or invalid [9].

8) **Privacy and security.** Storing or accumulating personal data over time raises privacy and security issues; it is possible that a retrievable memory store could leak personal information in case of lack of security.

9) **Benchmarking.** There are very few datasets that will test memory ability over many sessions

XI. FUTURE DIRECTIONS

Adaptive self-managing memory. A system need to learn what to write and merger or delete by getting feedback thus requiring less training compare to current existing RL systems [4].

Memory compression Need to be carried out in hierarchical abstractions and special learned compressions to decrease the amount of memory usage.

Privacy-preserving memory Memory should be kept user-controlled.

Multimodal lifelong memory Storage and linking of text, image, audio, video and sensors data for embodied and assistive agents.

Single memory hierarchy. The context window and external store should be incorporated into a single hierarchical memory instead of separate modules.

Standard benchmarks and metrics. Using common metrics for assessment: accuracy, latency, cost, and safety.

Hybrid memory system Combination of simple external updates and infrequent editing of weights for stable knowledge.

A. Limitations of This Study

This paper presents a literature review with experiments involved. The comparison in Table III is merely a qualitative reading of the descriptions of various pieces of work written by authors and some of the 2026 works are in preprint form and may have changed before being published. Any statement should be viewed together with the original paper.

XII. CONCLUSION

The employed LLM systems are able to practice lifelong memory that allows them to use its knowledge for the following interactions. Studies indicate development from mere storage of text to thinking over it to independent operation with text and its abstraction to reusable knowledge. Efficient design depends on incorporation of writing, retrieving, updating, compacting and forgetting processes. Nonetheless, significant obstacles remain, including scalability issues, memory conflicts, high computational costs, and privacy concerns, as well as the lack of common architectures and benchmarks. Improvements in these fields would result in more personalized, consistent, and adaptable generative AI systems.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

References

[1] S. Zhoubian, D. Zhang, E. Kharlamov, and J. Tang, "Memory for large language models," arXiv preprint arXiv:2607.25380, Jul. 2026.

[2] J. Luo, Y. Tian, C. Cao, Z. Luo, H. Lin, K. Li, C. Kong, R. Yang, and J. Ma, "From storage to experience: A survey on the evolution of LLM agent memory mechanisms," in *Findings Assoc. Comput. Linguistics: ACL 2026*, Jul. 2026, pp. 41622–41652.

[3] J. Zheng, C. Shi, X. Cai, Q. Li, D. Zhang, C. Li, D. Yu, and Q. Ma, "Lifelong learning of large language model based agents: A roadmap," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 48, no. 5, pp. 5552–5571, May 2026, doi: 10.1109/TPAMI.2025.3650546.

[4] Y. Yu, L. Yao, Y. Xie, Q. Tan, J. Feng, Y. Li, and L. Wu, "Agentic memory: Learning unified long-term and short-term memory management for large language model agents," arXiv preprint arXiv:2601.01885, Jan. 2026.

[5] J. Kang, M. Ji, Z. Zhao, and T. Bai, "Memory OS of AI agent," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2025, pp. 25961–25970.

[6] P. Chhikara, D. Khant, S. Aryan, T. Singh, and D. Yadav, "Mem0: Building production-ready AI agents with scalable long-term memory," arXiv preprint arXiv:2504.19413, Apr. 2025.

[7] Y. Wu, S. Liang, C. Zhang, Y. Wang, Y. Zhang, H. Guo, R. Tang, and Y. Liu, "From human memory to AI memory: A survey on memory mechanisms in the era of LLMs," arXiv preprint arXiv:2504.15965, Apr. 2025.

[8] W. Xu, Z. Liang, K. Mei, H. Gao, J. Tan, and Y. Zhang, "A-MEM: Agentic memory for LLM agents," arXiv preprint arXiv:2502.12110, Feb. 2025.

[9] P. Rasmussen, P. Paliychuk, T. Beauvais, J. Ryan, and D. Chalef, "Zep: A temporal knowledge graph architecture for agent memory," arXiv preprint arXiv:2501.13956, Jan. 2025.

[10] B. J. Gutiérrez, Y. Shu, Y. Gu, M. Yasunaga, and Y. Su, "HippoRAG: Neurobiologically inspired long-term memory for large language models," in *Advances Neural Inf. Process. Syst. (NeurIPS)*, vol. 37, 2024.

[11] A. Maharana, D.-H. Lee, S. Tulyakov, M. Bansal, F. Barbieri, and Y. Fang, "Evaluating very long-term conversational memory of LLM agents," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics (ACL)*, 2024, pp. 13851–13870.

[12] W. Zhong, L. Guo, Q. Gao, H. Ye, and Y. Wang, "MemoryBank: Enhancing large language models with long-term memory," in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 17, 2024, pp. 19724–19732.

[13] J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, "Generative agents: Interactive simulacra of human behavior," in *Proc. 36th Annu. ACM Symp. User Interface Softw. Technol. (UIST)*, 2023, pp. 1–22.

[14] C. Packer, S. Wooders, K. Lin, V. Fang, S. G. Patil, I. Stoica, and J. E. Gonzalez, "MemGPT: Towards LLMs as operating systems," arXiv preprint arXiv:2310.08560, Oct. 2023.

[15] W. Wang, L. Dong, H. Cheng, X. Liu, X. Yan, J. Gao, and F. Wei, "Augmenting language models with long-term memory," arXiv preprint arXiv:2306.07174, Jun. 2023.