



On-Device Agent Memory: Forms, Lifecycle Management, and Personalization

Wenqi Hu¹, Yuxiang Ren^{1,*}, Ran Zhou¹, Qijia Zhuang¹, Zihan Shen¹, Yanyin Gou¹, Zehui Kong¹, Fanyu Lin², Jiale Chang³, Zhen Zhang¹, Jiaheng Liu¹, Haopeng Zhang⁴, Jiawei Zhang⁵, Philip S. Yu⁶

¹Nanjing University, Nanjing, China

²Peking University, Beijing, China

³Nanjing Agricultural University, Nanjing, China

⁴University of North Carolina at Charlotte, Charlotte, North Carolina, USA

⁵University of California, Davis, Davis, California, USA

⁶University of Illinois Chicago, Chicago, Illinois, USA

*Corresponding author

Memory is central to the ability of agents built on large language models to maintain task continuity and adapt to individual users. On personal devices, these capabilities draw on multimodal observations and interaction histories accumulated during everyday use. As histories grow and user needs evolve, memory systems face interrelated challenges in maintaining useful information under resource constraints while preserving user control. Understanding these challenges calls for a perspective that connects memory design with the practical requirements of on-device use. This survey examines on-device agent memory through the complementary perspectives of memory forms and lifecycle management. We then consider how these design choices support personalization. Furthermore, we examine current progress in deployment and evaluation, and discuss the broader challenges shaping the future of on-device agent memory. Looking ahead, we hope this survey serves as a reference for existing research and a foundation for developing reliable on-device agent memory.

✉ wenqihu@mail.nju.edu.cn, renyuxiang@nju.edu.cn

🌐 <https://github.com/holmeshu/On-Device-Agent-Memory-Survey>

🌐 <https://holmeshu.github.io/On-Device-Agent-Memory-Survey>

1 INTRODUCTION

Large language models (LLMs) increasingly underpin agents that use tools, carry out multistep tasks, and interact with their environments over time [150, 186]. These agents typically combine language modeling with reasoning, planning, perception, memory, and tool use [139, 155, 174]. Memory is central to their ability to accumulate experience and adapt through continued interaction [57, 200]. As tasks unfold across interactions, decisions often depend on earlier observations and feedback. By retaining, organizing, and accessing this history, memory connects successive stages of reasoning and action, allowing agents to maintain task state and reuse experience [56, 131].

Personal devices bring this dependence on history into everyday use. Smartphones, computers, and wearables supply observations that accumulate into a user’s history and may need to be retained and accessed repeatedly [166, 183, 190]. This history can support personalization by helping an agent interpret current requests in light of the user’s preferences and past experiences [125, 209]. When this history is accessed and processed in the cloud, repeated interactions incur

communication and remote processing costs [212]. Running inference and memory processing on the device can reduce transfers of sensitive information and support offline operation [160]. However, local execution places memory storage and maintenance alongside model inference and other applications within a shared, limited resource budget [190, 202]. Larger histories and model contexts generally increase the costs of storage, access, and inference [111]. The challenge for on-device agent memory is to retain and access useful history for task execution and personalization within these resource limits. Figure 1-1 summarizes three characteristics that shape this problem relative to common cloud deployments.

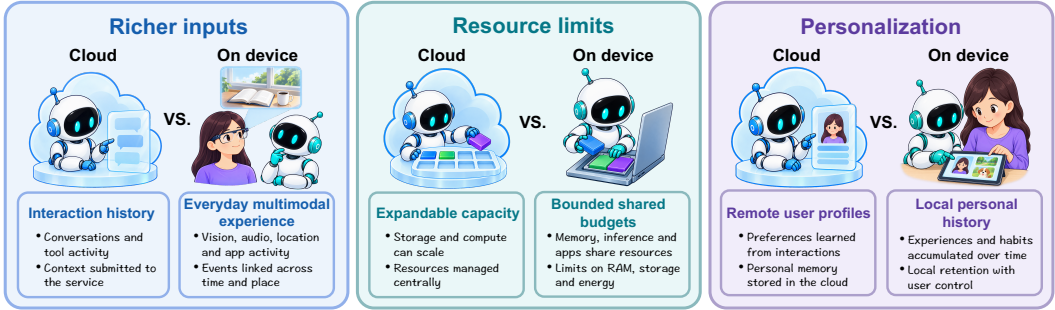


Fig. 1-1. Comparison of typical cloud and personal device deployments of agent memory across inputs, resource budgets, and personalization.

- Richer inputs: from discrete interactions to continuous multimodal experience.** Question-answering and tool-use agents are often studied within discrete tasks, with histories organized around a particular interaction or episode [56, 186]. Personal devices can also capture ongoing visual, audio, location, and interaction data, extending the input to multimodal experience linked across time and space [166, 183, 215]. Extracting events and semantic cues from these streams turns scattered observations into history that can be accessed later [183, 215]. Repeated content and transient changes make selection a central part of memory formation. Memory formation therefore involves balancing the value of retained information and context against the cost of continual processing [112, 190].
- Resource limits: from scalable capacity to bounded memory and energy budgets.** Cloud systems can expand capacity and schedule resources centrally to accommodate larger indexes and contexts. On personal devices, memory operations share fixed hardware with model inference and foreground applications [160, 190, 202]. Available runtime memory limits how much information can remain loaded. Loading on demand and tiered storage adjust the resident footprint and data movement costs [4, 111]. Both immediate memory use and longer-term maintenance draw on the same resource and energy budgets [111, 202].
- Personalization: from general knowledge to a user's long-term history.** In general-purpose tasks, memory can supplement public knowledge and execution experience [139, 200]. Personal devices also offer a setting for accumulating a particular user's interaction history over time. Organizing this history can help an agent understand user preferences and interpret current tasks in light of personal context [125, 209]. Personalization can draw on contextual details of individual experiences and relatively stable information across events [183, 209]. Updating this information helps memory remain relevant as users and environments change [166, 179]. As personal information accumulates, the memory system should give users control over its retention and use [10, 166].

These characteristics call for a closer look at memory in sustained use on personal devices. As personal histories grow, keeping them useful takes ongoing work within a limited budget. What an agent retains also depends on the user’s needs and the permissions governing later use. This survey examines existing memory research with these conditions in view.

We begin with the concepts and constraints used throughout the review. From there, we consider how memory is represented and maintained, and how it supports personalized services. Memory design and intended use shape one another throughout this discussion. The chosen representation affects how information is accessed and kept up to date. At the same time, personalization goals and privacy requirements guide what the agent retains. We then examine these choices in the device settings where they have been studied. Evaluation helps assess the capabilities demonstrated and the costs of sustaining them. The contributions below follow this progression and indicate the corresponding sections:

- (1) **Foundations (Section 2).** We formalize how agent memory is constructed and used, first in general and then under on-device constraints. Cognitive concepts clarify differences in retention timescale and function. Together with lifecycle operations, resource budgets, and data-use rules, these definitions establish a common basis for comparing methods.
- (2) **Memory Forms (Section 3).** We organize methods by their basic memory units into five forms: textual, vector, key–value (KV) cache, graph-structured, and parametric memory. Comparing their uses and combinations clarifies the distinct roles of memory content, access structures, and storage location.
- (3) **Lifecycle Management (Section 4).** Formation, retrieval, evolution, and forgetting provide the structure for examining memory management during continued interaction. We analyze how writes, access, and maintenance affect one another under device constraints, exposing trade-offs between memory effectiveness and maintenance cost.
- (4) **Personalization (Section 5).** We examine how personal history informs preference learning, personal reference resolution, and proactive assistance. Across these uses, privacy safeguards govern access, retention, and disclosure, linking useful personalization to users’ control over their information.
- (5) **Device Deployment (Section 6).** Deployment evidence is organized across personal computing, head-worn, personal sensing, and embodied systems. For each context, we examine component execution locations, experimental configurations, and reported resource measurements. This assessment distinguishes demonstrated local execution from the evidence available to characterize its resource demands.
- (6) **Evaluation (Section 7).** We organize evaluation around memory accuracy, lifecycle management, personalization quality, and on-device runtime cost. General agent-memory benchmarks and device-context benchmarks are compared by their tasks and measurement scope. This comparison identifies their complementary coverage and the evidence still needed to evaluate sustained on-device operation.

Figure 1-2 provides an overview of the survey and representative works discussed in each part.

2 FOUNDATIONS OF ON-DEVICE AGENT MEMORY

2.1 Agent Memory

Effective decision-making often requires information beyond an agent’s current observation [57, 139]. Memory can preserve what remains relevant from an ongoing task and make experience from earlier tasks available again [56, 200]. We express this role through an evolving memory state and the decision policy that uses it [57].

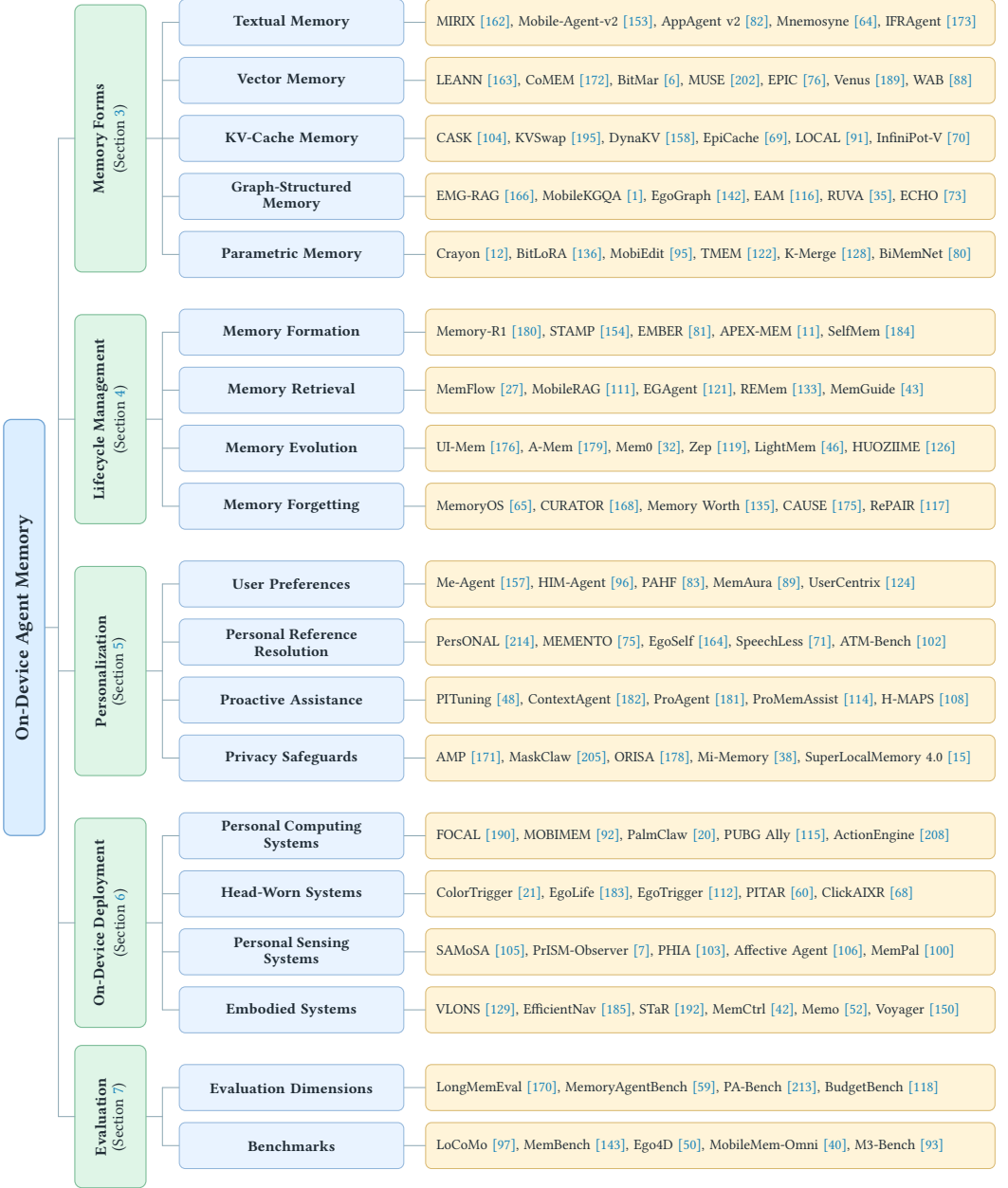


Fig. 1-2. Taxonomy of on-device agent memory. The survey is organized around memory forms (Section 3), lifecycle management (Section 4), personalization (Section 5), deployment contexts (Section 6), and evaluation (Section 7). Each row lists representative works discussed in the corresponding subsection.

Definition 2.1 (Agent Memory). Agent memory is an evolving state M_t through which information from previous observations and interactions remains available to the agent’s decision process. Let $H_{<t}$ denote the history of observations, actions, and feedback before time t . Memory construction and use are expressed as

$$M_t = \mathcal{G}(H_{<t}),$$

$$a_t = \pi(x_t, M_t).$$

Here, $\mathcal{G}$ selects, organizes, and maintains information from $H_{<t}$, while π denotes the agent's decision policy. The current request and its context are represented by x_t , and a_t is the resulting response or action.

Memory can retain the details of a particular experience or knowledge that remains useful across different situations [110]. These contents also differ in how long they remain available. Cognitive research offers a vocabulary for separating these aspects, beginning with short-term and long-term retention [36].

- **Short-term memory.** Short-term memory temporarily holds a limited amount of information for an ongoing activity [36]. Agent systems draw on this notion to describe recent observations and intermediate results kept available within the current task [56, 200]. Their immediate value lies in preserving the task's progress as it moves from one step to the next [56]. This continuity can span several model calls while remaining tied to the activity in progress [139].
- **Long-term memory.** Long-term memory preserves knowledge and experience beyond the activity in which they were acquired [36]. In agent systems, it can span multiple sessions and tasks, retaining information that is no longer part of the immediate interaction [200]. This persistence allows an agent to build on earlier experience, including an understanding of a user developed across encounters [209].

Short-term retention provides the information on which ongoing reasoning can operate [36]. **Working memory** emphasizes this active use, combining temporary retention with the processing needed to manipulate intermediate results and integrate new input with existing knowledge [9]. For an agent, a working state can bring together current observations and goals with knowledge drawn from long-term memory [139]. Its contents reflect what matters to the activity underway, so they can include older knowledge as well as recent input [139].

Within long-term memory, a further distinction concerns the content retained and the role it serves. Cognitive research distinguishes memory for events and facts from the learned skills involved in performing an activity [137]. Agent research draws on the terms episodic, semantic, and procedural memory to describe these different roles [139].

- **Episodic memory.** An episodic memory preserves a particular experience together with the circumstances that give it meaning [137]. In agent systems, it links observations and actions to the setting and outcome of a specific interaction [110, 139]. Temporal and contextual associations help distinguish similar experiences and locate them within a longer personal history [166, 183]. These associations make the event and its circumstances available when later requests concern what happened at a particular time or place [183].
- **Semantic memory.** Semantic memory holds facts and relationships whose use does not depend on recalling the episode in which they were acquired [137]. For an agent, this knowledge can concern its tasks and environment as well as the user it serves [139]. It may include information stated directly during interaction and broader inferences drawn from several experiences [110, 209].
- **Procedural memory.** Procedural memory concerns learned skills and ways of carrying out an activity [137]. Agent research adopts this functional analogy for reusable procedures that guide task execution [139, 167]. A procedure can capture the organization of actions needed to achieve a goal, allowing that pattern to be applied when the task's details change

[167]. Previously acquired skills can also be adapted or combined to support more complex activities [150].

These distinctions describe complementary aspects of memory. Episodic, semantic, and procedural knowledge can all persist over the long term, and information from these memories can become relevant to current processing [139]. Their roles can also coexist within the same history. A remembered interaction may remain available as an episode alongside the more general knowledge derived from it [110]. We use these cognitive concepts to describe memory functions without prescribing a separate storage module for each one. The same function may be supported by different representations, and one representation may serve several functions [57]. Section 3 examines these memory forms and how they can be combined.

2.2 On-Device Agent Memory

2.2.1 Formulation. On-device deployment brings the cost of memory construction and use into the problem definition. Processing and storage draw on limited device capacity, while energy availability constrains sustained operation [111, 160]. For personal agents, the use of historical information also depends on the purpose and authorized scope of an interaction. Information available to the system may be inappropriate to disclose in a particular task [10]. We therefore extend the general formulation by making resource budgets and data-use rules explicit.

Definition 2.2 (On-Device Agent Memory). On-device agent memory is agent memory retained and processed on a personal device under resource budgets $\mathbf{b}$ and data-use rules $\mathcal{P}$. Using the notation in Definition 2.1, its construction and use are expressed as

$$\begin{aligned} M_t &= \mathcal{G}(H_{<t}; \mathbf{b}, \mathcal{P}), \\ a_t &= \pi(x_t, M_t; \mathbf{b}, \mathcal{P}). \end{aligned}$$

The resource budgets $\mathbf{b}$ apply to local agent execution, including memory processing and model inference. The rules $\mathcal{P}$ specify which information may be retained and whether it may be accessed or disclosed for a given task. Only executions that satisfy the resource limits and data-use requirements specified in Section 2.2.3 are considered.

In this setting, memory processing and model inference share the resources allocated to the agent, so maintaining accumulated history can affect the capacity available for current decisions [111, 202]. Memory management continues across interactions as new observations arrive and the relevance of retained information changes [166, 179]. Section 2.2.2 describes this continuing process through the memory lifecycle.

2.2.2 Memory Lifecycle Definition. An agent draws on memory when deciding how to act, and the outcome of that action can provide new information to retain [139]. Feedback can also change the interpretation of an earlier experience or the value of keeping it [131, 179]. Memory use and revision therefore belong to the same continuing interaction with the environment [57]. Some systems support adaptation by retaining feedback and lessons from prior attempts while keeping the underlying language model fixed [131, 201].

We describe this process through four operations: formation, retrieval, evolution, and forgetting. Each operates under the resource budgets $\mathbf{b}$ and data-use rules $\mathcal{P}$. For operations that change the memory state, M'_t denotes the resulting state.

Memory formation. Formation selects new information with potential value for later use and organizes it into memory units [179, 180]. With o_t denoting incoming observations, interaction outcomes, and feedback, formation updates memory as

$$M'_t = \mathcal{F}(M_t, o_t; \mathbf{b}, \mathcal{P}).$$

The resulting units should preserve enough provenance and context to support later judgments about what the information means and when it applies [110, 166].

Memory retrieval. A decision usually depends on only part of the available history [110, 139]. Retrieval extracts a memory signal z_t from M_t according to the current request and context x_t :

$$z_t = \mathcal{R}(M_t, x_t; \mathbf{b}, \mathcal{P}).$$

Systems with explicit retrieval use z_t to inform the policy. The amount returned affects both the historical evidence available to the policy and the cost of subsequent inference [56, 111].

Memory evolution. As experience accumulates, stored information may become redundant or inconsistent [166, 179]. Let u_t denote new evidence, feedback, or consolidation requirements that motivate a revision:

$$M'_t = \mathcal{E}(M_t, u_t; \mathbf{b}, \mathcal{P}).$$

Formation introduces new information, whereas evolution consolidates and updates existing content and relationships. This allows scattered experience to be integrated and outdated knowledge to be revised [166, 179].

Memory forgetting. Changes in retention value or a user's deletion request can make existing information a target for forgetting [65, 166]. Writing d_t for the forgetting criteria or designated target, we express the operation as

$$M'_t = \mathcal{D}(M_t, d_t; \mathbf{b}, \mathcal{P}).$$

The aim is to reduce or remove that information's influence on later decisions while preserving the usefulness of other memories [65, 193].

The timing and frequency of these operations depend on task needs and device budgets [65, 202]. An interaction may involve only part of the lifecycle, and the operations need not follow a fixed sequence. Section 4 examines how existing methods organize these operations and coordinate them over continued interaction.

2.2.3 On-Device Constraints. On-device use of personal history depends on both the capacity to process information and authorization to retain and use it [10, 160]. Resource budgets $\mathbf{b}$ and data-use rules $\mathcal{P}$ express these two requirements. Memory processing and model inference share the resources allocated to the agent, while other applications affect how much the device can provide [111, 202]. For a given device and task configuration, let W denote the period of operation under consideration. The budget $\mathbf{b} = (B_R, B_S, B_T, B_E(W))$ constrains resource demands as follows:

$$R_{\text{peak}} \leq B_R, \quad S \leq B_S, \quad T_{\text{resp}} \leq B_T, \quad E(W) \leq B_E(W).$$

Available runtime memory limits how much the agent can hold at once. Reading and processing stored information consume part of the same memory space needed for model inference [111, 202]. Here, R_{peak} measures the peak runtime memory footprint of the agent as a whole, bounded by B_R . Persistent storage preserves data across interactions. Here, S is the storage footprint of the specified configuration, and B_S is its limit. Response time and energy capture the demands of immediate interaction and sustained operation, respectively. T_{resp} measures the time from receipt of a request to return of its result, with B_T specifying the allowed maximum. Energy consumption $E(W)$ includes both interactive processing and background maintenance performed during W , subject to the budget $B_E(W)$. Maintenance can be scheduled separately from latency-sensitive

queries when it is not needed for the current response [202]. Its energy use still counts toward the budget of the interval in which it runs. The four resource dimensions also guide the audit in Section 6.

Data-use rules $\mathcal{P}$ define which information the user authorizes the agent to retain, for how long, and for which tasks. Under these rules, later use remains subject to access permissions, while disclosure to tools or remote services must be limited by task purpose and recipient [10]. User requests to correct or delete stored information must be implemented through memory updating and forgetting [166].

3 MEMORY FORMS IN ON-DEVICE AGENTS

As a starting point for organizing prior work, we examine the fundamental representational units that carry memory in on-device agents. The central question is what forms on-device agent memory can take.

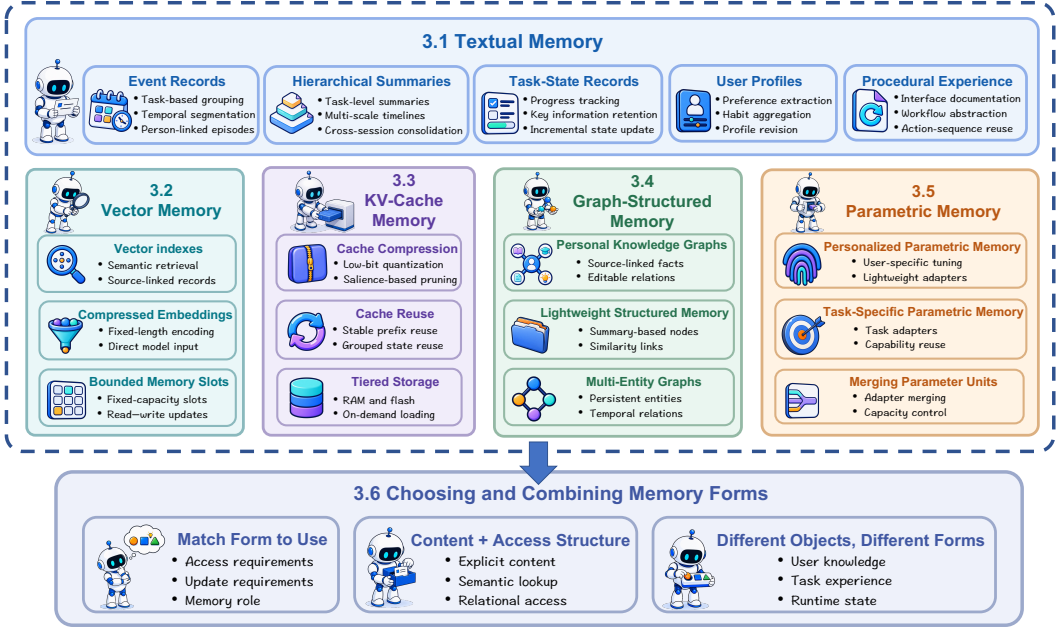


Fig. 3-1. Five memory forms in on-device agents and the subcategories used to organize Chapter 3.

A single experience may be retained as a human-readable record or transformed into continuous representations, relational structures, or parameter changes. These forms are not interchangeable encodings of the same content. They preserve information at different levels of granularity and expose different units for direct access. In on-device settings, the choice of memory form shapes how readily users can inspect and revise memory. It also affects the resource cost of maintaining and using memory over time.

Based on the primary memory unit retained and retrieved by each system, we organize prior work into five forms: textual memory (Section 3.1), vector memory (Section 3.2), KV-cache memory (Section 3.3), graph-structured memory (Section 3.4), and parametric memory (Section 3.5). This taxonomy establishes a structural framework for understanding what carries memory in on-device agents. Each form gives past information a different structure and a different route into later

reasoning. Section 3.6 compares their roles and management considerations and examines how multiple forms can be combined in practical systems. Figure 3-1 provides an overview of these five forms and the subcategories used to organize the discussion in Sections 3.1–3.5. Table 3-1 summarizes the memory forms and structures of representative methods discussed in this chapter.

Table 3-1. Memory forms and structures of representative methods

Method	Memory Form	Memory Structure
FOCAL [190]	Textual	Task logs with summaries
MIRIX [162]	Textual	Typed personal memory records
EgoLife [183]	Textual	Timestamped events and layered summaries
AR Secretary Agent [44]	Textual	Conversation records linked to people
EMem / EMem-G [211]	Textual + Graph	Event records with an optional graph
Mnemosyne [64]	Textual + Graph	Summaries with associative links
HiAgent [56]	Textual	Subgoal summaries linked to actions
StructMem [177]	Textual	Summaries spanning multiple event records
CAM [78]	Textual + Graph	Overlapping clusters with layered summaries
HiGMem [23]	Textual	Event summaries linked to dialogue turns
HEMA [2]	Textual + Vector	Layered summaries with indexed dialogue
HyMem [203]	Textual + Vector	Indexed summaries linked to dialogue
Mobile-Agent-v2 [153]	Textual	Progress records retaining key information
adaptive context management [149]	Textual	Incrementally revised task state
PUBG Ally [115]	Textual + KV cache	Working plans with reusable context caches
TSR [207]	Textual	Instruction and progress records
Stamp-GUI [154]	Textual	Stepwise records of key information
ATMem [84]	Textual	Task state organized by dependency
LightWM [165]	Textual	Global state with subgoal records
HIM-Agent [96]	Textual	Hierarchical intent and behavior records
Me-Agent [157]	Textual	User profiles across and within applications
IFRAgent [173]	Textual	Procedure records with personal habits
ProAgent [181]	Textual	Evolving user profile entries
PersonaX [130]	Textual	Personas derived from behavior clusters
Bi-Mem [101]	Textual + Graph	Facts linked to summaries and profiles
QUMem [152]	Textual	User memories linked to source records
O-Mem [156]	Textual	User profiles with topic indexes
AppAgent [194]	Textual	Interface documentation for available actions
AppAgent v2 [82]	Textual + Vector	Interface documentation with vector indexes
UI-Mem [176]	Textual + Vector	Indexed workflow and failure templates
MOBIMEM [92]	Textual + Graph	Profile graph with procedural records
MAGNET [141]	Textual + Vector	Indexed workflows and interface examples
mobile action replay [94]	Textual	Successful requests with action sequences
PROJECTMEM [99]	Textual	Project event logs with failure records
MUSE [202]	Vector	Vector index with inverted lists
MobileRAG [111]	Vector	Partitioned index with partial loading
LEANN [163]	Vector	Graph index with regenerated embeddings
WAB [88]	Textual + Vector	Indexed browser records across storage tiers
Lever [41]	Vector	Local vector subindex shaped by queries
EPIC [76]	Textual + Vector	Textual records with instruction indexes
CoreMem [28]	Textual + Vector	Vector index over textual records
Venus [189]	Vector	Hierarchical index over video frames
STaR [192]	Vector	Scene records with spatiotemporal indexes
CoMEM [172]	Vector	Trajectory embeddings of fixed length
clustered memory compression [17]	Vector	Clustered representations of personal records

Table 3-1 (continued)

Method	Memory Form	Memory Structure
BitMar [6]	Vector	Quantized memory slots with fixed capacity
QuantSpec [146]	KV cache	Shared cache with hierarchical quantization
CASK [104]	KV cache	Selectively pruned and quantized cache
InfiniPot-V [70]	KV cache	Bounded cache of selected video tokens
EpiCache [69]	KV cache	Separate caches for dialogue topics
PolyKV [113]	KV cache	Shared pool with asymmetric precision
Don't Waste Bits! [18]	KV cache	Cache with adaptive token precision
Agent-X [34]	KV cache	Reusable prefixes for tool descriptions
EfficientNav [185]	KV cache	Separate caches for map groups
CIFLEX [77]	KV cache	Shared cache with subtask branches
KVFlow [109]	KV cache	Prefix cache tree for agent workflows
KVCOMM [188]	KV cache	Shared segments with alignment anchors
JMSRA [37]	KV cache	Shared cache of prior agent context
ReCache [47]	KV cache	Separate blocks for tools and skills
SparseX [198]	KV cache	Reusable segments recomputed selectively
Nexus [8]	KV cache	Tool prefixes with recomputed suffixes
LOCAL [91]	KV cache + Parametric	Versioned caches paired with local adapters
KVSwap [195]	KV cache	Disk cache with compact relevance records
DynaKV [158]	KV cache	Dynamic cache groups across storage tiers
TierKV [134]	KV cache	Exact and compressed cache tiers
persistent 4-bit KV caches [132]	KV cache	Persistent quantized caches for each agent
EAM [116]	Graph	GUI state graph with reusable action paths
EMG-RAG [166]	Graph	Personal graph linked to source records
user profile memory module [75]	Graph	User profile graph linked to trajectories
RUVA [35]	Graph	Editable graph of personal memories
ECHO [73]	Textual + Graph	Personal records with temporal links
MobileKGQA [1]	Vector + Graph	Knowledge graph with binary relation codes
SuperLocalMemory [13]	Graph	Hierarchical records with similarity links
EgoSelf [164]	Graph	Event graph with persistent entities
EGAgent [121]	Graph	Entity relations with validity intervals
EgoGraph [142]	Graph	Temporal graph tracking entity histories
PITuning [48]	Parametric	Predictor parameters adapted to users
CDCDA-PLM [210]	Parametric	Separate LoRA adapters for individual users
EcoTune [26]	Parametric	Selected model layers adapted to interactions
BitLoRA [136]	Parametric	LoRA adapters over quantized parameters
MobiEdit [95]	Parametric	Localized weights storing personal facts
Crayon [12]	Parametric	Task adapters blended from existing modules
BiMemNet [80]	Parametric	Separate parameter pathways for tasks
TMEM [122]	Parametric	LoRA weights retaining task experience
PEAM [51]	Parametric	Separate LoRA adapters for skill categories
DuoMem [54]	Textual + Parametric	Procedural records with LoRA adapters
HydraOpt [24]	Parametric	Shared and separate task parameters
K-Merge [128]	Parametric	Bounded pool of merged task adapters

3.1 Textual Memory

Among the five memory forms introduced above, textual memory stores selected information from prior experience in natural language or structured text. Since the stored content remains explicit, it can be inspected and revised directly. This property is particularly useful when an on-device agent needs to retain personal information over the long term. The following subsections examine

several common categories of textual memory in on-device agents, distinguished by the content they retain and their level of abstraction.

3.1.1 Event Records. Continuous sensing presents an on-device agent with an ever-growing stream of activity. Its temporal order does not reliably align with task boundaries. Adjacent observations may belong to different tasks, while fragments separated in time may still form part of the same activity. One line of work organizes such inputs into event records associated with temporal, task, or person-specific context, allowing past experience to be located later. FOCAL [190] targets continuous desktop activity in which multiple tasks are interleaved. It first discards invalid actions and assigns the remaining activity to tasks using application and window metadata. It then writes descriptions of selected interface changes to task-specific local records. Rather than mixing all desktop activity into a single timeline, this design preserves task identity when each record is created. Addressing a similar stream of screen activity, MIRIX [162] filters redundant frames before routing the retained information to different types of personal memory. It stores content about past experiences as event entries carrying temporal and contextual information. Continuous input may also come from first-person audiovisual recordings. EgoLife [183] segments such streams into temporal intervals and creates a timestamped event description for each interval. The resulting record retains people, objects, and activities as events within a personal history. The record also includes third-person views and speaker-labeled transcripts. These help analyze interpersonal interactions and social dynamics. AR Secretary Agent [44] identifies the person with whom the user is speaking and stores the associated conversational content. When that person is encountered again, the agent retrieves and displays the previously stored summary of the earlier meeting. EMem and EMem-G [211] similarly organize memory into event-like propositions based on elementary discourse units (EDUs). MEMARENA [196] further brings persistent event logging to on-device settings, where textual records remain the primary memory form.

The defining issue for an event record is not its textual length, but the continuity used to determine what belongs to the same event. Tasks, time, and people can all serve as event anchors, allowing later retrieval to reconstruct an activity around different forms of context.

3.1.2 Hierarchical Summaries. As the span of recorded experience grows, a single event may no longer capture an entire task or extended experience. Some systems therefore construct hierarchical summaries over event records. These higher-level textual units are typically linked to lower-level records by task scope or temporal span. Building on its task-specific event logs, FOCAL [190] further summarizes the records associated with each task. This preserves task structure under interruptions and allows interrupted activities to be reviewed on a per-task basis. EgoLife [183] starts from timestamped event descriptions and successively produces hourly and daily accounts, making experience across multiple days accessible at different temporal scales. Both methods derive summaries from event records, preserving task structure and temporal structure, respectively.

A different form of hierarchy emerges incrementally over long-term interaction without aggregating records along predefined task or that temporal boundaries. Mnemosyne [64] takes interaction summaries as input, then removes entries that contain little long-term value or duplicate existing memory. The retained summaries form a lower tier of long-term memory. The system periodically extracts more stable user attributes and domain information from this tier to produce a cross-session overview. Unlike FOCAL and EgoLife, Mnemosyne builds its overview from summaries that have already been selected for long-term retention. Beyond these task- or time-anchored hierarchies, researchers are also exploring hierarchical summarization with different organizational units, such as subgoal-based memory in HiAgent [56], cross-event synthesis in StructMem [177], and incremental overlapping clustering in CAM [78]. Additionally, hierarchical designs have been developed with event-turn anchors in HiGMem [23] and two-level summary hierarchies in HEMA [2].

3.1.3 Task-State Records. Not all textual memory is intended for later recall. For an unfinished task, some systems retain the information required to continue execution and rewrite it as the task progresses. For mobile interaction, Mobile-Agent-v2 [153] condenses a lengthy action history into a record of current task progress. It separately retains key information from earlier screens that will still be needed in later steps. The progress record captures what has already been completed, helping the agent determine what remains to be done. The short-term information record carries content that may disappear from the current screen, avoiding the need to revisit earlier pages during cross-application tasks. The two records preserve execution progress and task-specific data in place of the complete multimodal interaction history.

Tasks involving dialogue and tool use can also produce long histories, for which a continuously updated task state can replace the full trajectory. An adaptive context-management framework [149] maintains a compact structured state instead of appending every interaction to the input. It updates this state after each turn, retaining the goals, constraints, and unresolved issues that continue to affect the current task. Each new interaction is converted into a state update, and history that no longer matters is omitted from subsequent model calls. PUBG Ally [115] maintains a similar working state for a game teammate running on a consumer PC. It refreshes transient game observations between agent loops while carrying unfinished commitments forward in compact plans. In both systems, memory preserves what is needed to continue the task without retaining the full trajectory.

Recent work on mobile graphical user interfaces has further refined what should be retained in a task state. TSR [207] first compares the interface before and after each action. It continually updates the initial task, completed subgoals, and current progress, separating relatively stable task state from transient screen observations. STAMP [154] constructs verifiable memory supervision in controllable virtual mobile environments, specifying what must be retained and when it will be needed. It trains Stamp-GUI to generate explicit textual memory during task execution. ATMem [84] further organizes content relevant to the task into a continuously updated hierarchical execution state, allowing task progress, the current availability of information, and the remaining actions to evolve together. LightWM [165] decomposes procedure-centric tasks into subgoals and organizes memory into task-level global memory and subtask-level local memory, where local memory directly conditions action selection and is updated through structured updates.

Across these systems, task-state memory retains the parts of an interaction history that remain relevant to execution and revises them as the task progresses. The resulting record may separate completed progress, currently needed information, and remaining actions without preserving the full trajectory.

3.1.4 User Profiles. Long-term personalization requires an on-device agent to distinguish one-off choices from user characteristics that persist across tasks. Related work generally looks for recurring evidence across multiple interactions and organizes it as preferences, habits, or user attributes. The Hierarchical Intent Memory Agent (HIM-Agent) [96] aggregates mobile interaction records over time and identifies recurring intentions and representative operations among similar activities. It combines frequency with behavioral consistency to derive general preferences and daily routines. Me-Agent [157] similarly infers user habits from mobile task experience, but further separates stable tendencies that recur across applications from preferences confined to a particular application. The key issue in profile construction is the scope over which each inferred characteristic remains valid. A choice observed within one application may indicate a scoped habit that does not necessarily generalize to all tasks.

User demonstrations can also reveal both general task procedures and personal choices. IFRAgent [173] extracts operation steps from each user’s mobile demonstrations. It separately identifies

personal habits from choices made at particular steps. The resulting records allow general procedures and user-specific tendencies to inform later tasks independently. When profiling draws on everyday sensing trajectories, ProAgent [181] infers user characteristics from long-term experiences collected by mobile and wearable devices. It then revises the entries as new experiences arrive during continued use. Information about the current environment is used to select profile content relevant to the present situation. Other works have also explored structured textual profiles. PersonaX [130] constructs offline textual personas from sub-behavior sequences, Bi-Mem [101] builds hierarchical fact-, scene-, and persona-level memory, QUMem [152] decomposes interaction histories into factual, preference, and insight memories, and O-Mem [156] dynamically updates user characteristics from proactive interactions.

3.1.5 Procedural Experience. For recurring interface tasks, merely recording what happened does not by itself reduce the reasoning required the next time the task is performed. Some studies therefore transform interface exploration, user demonstrations, or completed task trajectories into procedural experience that can guide subsequent actions. During autonomous exploration or observation of user demonstrations, AppAgent [194] compares the interface before and after an operation and documents interface controls, available actions, and their effects. These records remain close to the concrete interface and primarily describe which actions individual controls support. AppAgent v2 [82] expands the range of interface information captured in this documentation. It organizes knowledge acquired from autonomous exploration and human operation into updatable structured documents. The resulting documents allow both sources of interface knowledge to support later actions.

Reusing experience for a complete task requires more than knowing the functions of individual interface controls. UI-Mem [176] derives overall task workflows and steps from successful trajectories for reuse across tasks. It also extracts conditions that may interrupt execution from failed trajectories. Specific names, values, and operation targets in the original trajectories are replaced with placeholders. These placeholders can be filled for a new instance, allowing the same structure to support tasks with different parameters.

Higher-level abstraction does not entirely displace fine-grained action sequences that can be reused more directly. MOBIMEM [92] stores task templates that can be filled for a new task separately from detailed interaction sequences. The two forms retain general task logic and action experience for similar interface states. When several successful trajectories exist for the same goal, MAGNET [141] derives their common operation stages while preserving alternative execution paths. As a result, an interface change need not invalidate the entire body of procedural experience.

Related systems retain successful mobile action sequences for replay [94] and project-specific failure records in PROJECTMEM [99] to guide later actions.

These representations retain different aspects of procedural experience at varying levels of abstraction. Interface documentation describes available operations, task templates capture reusable task structure, action sequences retain concrete steps, and failure records identify situations that warrant caution. Their value depends on how the retained experience supports the next task.

3.2 Vector Memory

Vector memory encodes text, images, or interaction trajectories into numerical vectors, enabling memory from different sources to be organized in a shared representation space. In on-device agents, these vectors may serve as indexes to explicit records or as memory units consumed directly by the model.

3.2.1 Vector Indexes. In on-device agents, interface knowledge and task experience often remain as readable documents or templates. Vector indexes link the current interface or task instruction to the

relevant records. AppAgent v2 [82] embeds interface elements and their functional descriptions in a vector store. It then retrieves the required documentation using identifiers and text observed on the current interface. UI-Mem [176] builds semantic indexes over task templates and fine-grained action experience, allowing reusable content to be retrieved for new tasks. MAGNET [141] matches task descriptions to workflows and functional descriptions to examples of interface elements, enabling semantic retrieval to support both task planning and action grounding.

Long-term indexes that remain on the device also incur storage and maintenance costs within a fixed resource budget. MUSE [202] treats the vector database as a smartphone memory subsystem and coordinates queries, writes, and index rebuilding across heterogeneous processors. MobileRAG [111] partitions the index and loads only the portions required for each query, reducing memory use during retrieval. LEANN [163] retains a compressed approximate index and recomputes exact embeddings only for the small set of candidates encountered during search, thereby reducing persistent storage requirements. Related designs include on-demand loading of browser-history snippets in WAB [88] and history-adapted device-side sub-indexes for cloud retrieval in Lever [41]. Together, these designs manage the storage and access costs of retained context. EPIC [76] is complementary. It reduces what must be stored in the first place. It aligns on-device memory construction with user preferences and keeps only preference-relevant content.

When vector indexes are applied to continuously generated video or robotic observations, preserving their links to the source records becomes particularly important. Venus [189] selects representative frames from a video stream on an edge device and encodes them as index vectors. Each vector remains linked to the corresponding cluster of source frames. STaR [192] stores encoded scene descriptions together with timestamps, poses, and source text in a vector database, then uses retrieved entries to locate keyframes and geometric records. In these systems, vectors narrow the span of history that must be examined. However, precise temporal localization and spatial reasoning still depend on links to the underlying evidence.

3.2.2 Compressed Embeddings. Other work treats vectors as compressed memory representations that can be consumed directly by the model. CoMEM [172] first retrieves GUI trajectories relevant to the current interface. It then compresses the screenshots and action sequence in each trajectory into a fixed-length sequence of continuous embeddings and supplies them directly to the vision-language model. Performance with continuous memory continues to improve as the memory collection and the number of retrieved trajectories increase. By contrast, text-based memory begins to degrade when too many entries are included. By compressing each trajectory independently, CoMEM bounds the input cost of each additional memory. The separation between trajectories is retained, allowing more historical trajectories to be used together.

A clustering-driven memory compression method [17] compresses representations across multiple personal records. It encodes each retrieved record as a continuous memory unit, groups the units by semantic similarity, and merges the representations within each group. Retaining several cluster-level representations outperforms collapsing all records into a single mean representation. A single global average removes distinctions between semantically different groups. Merging within clusters is designed to retain these distinctions. The fixed capacity prevents resident memory from growing linearly with interaction history. Meanwhile, other work focuses on retrieval and compression under edge budgets, such as CoreMem [28].

3.2.3 Bounded Memory Slots. A fixed-capacity read–write structure offers another way to prevent vector memory from expanding as new inputs continue to arrive. BitMar [6] uses low-bit encoders to produce textual and visual vectors. It reads and writes multimodal context through a fixed number of key–value slots and passes the memory readout to the decoding process. Unlike systems that retain an independent vector for each experience, BitMar updates a shared set of slots whenever

new information is written. The fixed capacity prevents resident memory from growing linearly with interaction history.

3.3 KV-Cache Memory

On-device agents often revisit tool descriptions, task histories, and environmental observations across successive planning steps and repeated invocations. KV-cache memory preserves the key and value states already computed for this context. Subsequent inference can reuse them instead of prefilling the entire input again. Cached states are tied to a particular model configuration and parameter version, and their footprint grows with the retained context. They also remain reusable only while subsequent inputs preserve a compatible prefix or segment structure. These constraints make cache growth and reusable segment boundaries central design concerns for on-device systems.

3.3.1 Cache Compression. When the full cache no longer fits in device memory, the system can lower the numerical precision of retained states. Alternatively, it can reduce the set of states available to subsequent attention. QuantSpec [146] follows the former route with a hierarchical 4-bit KV cache for self-speculative decoding. The draft and target computations share this cache instead of maintaining a separate full cache for each. CASK [104] combines quantization with pruning. It estimates the salience of KV pairs from recency, access frequency, attention allocation, and gradient attribution. Based on these scores, it quantizes states with moderately low salience and removes lower-scoring states. Both methods reduce the cache footprint, but they alter different aspects of the retained information. Quantization introduces numerical approximation while largely preserving the accessible token history, whereas pruning narrows the history available to later attention.

Growing histories also require retention decisions before the next query is known. InfiniPot-V [70] maintains a bounded KV cache for streaming video by removing temporally redundant tokens and retaining tokens with large value norms as new frames arrive. EpiCache [69] addresses long conversations by constructing several topic-conditioned KV caches through block-wise prefill, which limits cache growth during context processing. It loads the cache matched to each new question while retaining the other topic caches in host memory. InfiniPot-V selects a shared visual history, whereas EpiCache preserves different views of the same dialogue for later routing.

Compression can also be combined with persistence and sharing across agents. One approach stores persistent 4-bit KV caches on disk for edge multi-agent inference [132], whereas PolyKV [113] shares an asymmetrically compressed KV pool across agents. Adaptive precision provides another degree of control, as in Don't Waste Bits! [18], which learns token-level KV quantization.

3.3.2 Cache Reuse. KV-cache reuse depends on whether recurring context can be exposed as a stable unit of computation. Prompts for tool-using on-device agents typically combine relatively stable tool information with task-specific inputs. Agent-X [34] rearranges these components so that tool descriptions and usage guidelines form an early prefix, and precomputes KV caches for selected tool combinations. At runtime, the system loads a cache that matches the current task before processing the remaining user instruction and environmental observations. This design shifts part of online prefill to precomputation and local loading.

When context evolves with new observations, the useful unit of reuse may need to be smaller than a complete prefix. EfficientNav [185] divides navigation map descriptions into groups and computes a separate KV cache for each group. New observations update only the affected groups, whereas planning loads the groups relevant to the current goal. Independent computation avoids repeatedly processing unchanged map content but removes attention interactions across groups. EfficientNav mitigates this loss by using attention signals to place strongly related content in the same group. This group-level decoupling exemplifies a broader trend in agentic KV cache reuse.

CIFLEX [77] extends reuse from token level to task level for on-device multi-turn interaction. It reuses the main-task KV cache and injects sub-task instructions into isolated side paths, avoiding redundant prefill when switching between tasks. Workflow-aware eviction and prefetching in KVFlow [109] and cross-context alignment in KVCOMM [188] address complementary reuse costs. For wireless collaboration, JMSRA [37] jointly selects token or KV-cache transmission and allocates communication resources. Tool and segment reuse is further explored through composition-invariant blocks in ReCache [47], non-prefix sharing in SparseX [198], and depth-dependent suffix recomputation in Nexus [8].

Reuse also depends on changes to the model that produced the cache. LOCAL [91] ties KV validity to both the input prefix and adapter version, and refreshes frequently used prefixes when local adapters are updated. It coordinates this refresh with foreground inference and background training, keeping reusable inference state consistent with the updated model.

3.3.3 Tiered Storage. Compression and reuse do not guarantee that all cached states can remain in device memory. Mobile and embedded systems commonly use shared memory, making local storage an additional capacity tier. KVSwap [195] retains the full KV cache in local storage and keeps a compact representation in memory to predict relevant cache groups. It then prefetches the required groups according to the access characteristics of the storage device. As decoding progresses, fixed groups may become misaligned with the evolving distribution of KV states, causing essential entries to be missed or redundant entries to be loaded. DynaKV [158] responds by adapting cache groups as access patterns change and reorganizing data across smartphone memory and flash so that related states are stored contiguously. In both cases, the retained memory remains the KV state itself, whereas the compact representations and indexes determine which states are brought into memory.

Tiered storage can also extend cache lifetime beyond a single continuous inference session. A persistent KV-cache approach [132] stores a separate 4-bit cache for each agent in local storage. It restores the cache directly to the attention layers when that agent is invoked again. Interleaved multi-agent execution also creates opportunities to overlap one agent’s cache loading with another agent’s decoding. Persistence therefore replaces repeated prefill with local I/O. TierKV [134] instead predicts future cache demand from prefill hidden states. It assigns tokens to exact, low-rank, and flash-offloaded tiers before decoding begins.

3.4 Graph-Structured Memory

Textual and vector memory can store and retrieve individual entries. When tasks depend on multiple personal facts, events separated in time, or relations among entities, the links between entries often have to be reconstructed after retrieval. Graph-structured memory makes these links explicit by treating entities, events, states, or complete memory entries as addressable nodes. It stores their connections as edges, so retrieval can use both individual entries and the structure linking them. For on-device agents that accumulate personal data over time, graph design determines which relationships remain available for later reasoning or action.

Graph structure can also retain how earlier actions changed an interface. Executable Agentic Memory (EAM) [116] stores explored GUI states and action transitions in an executable graph, grouping recurring action sequences into reusable units. A small value model guides tree search over this graph, and a cloud model selects and parameterizes the resulting paths. Here, edges preserve executable transitions, complementing the factual and associative relations considered below.

3.4.1 Personal Knowledge Graphs. A personal knowledge graph first links records scattered across a device to the same user and the relevant entities. EMG-RAG [166] addresses this problem using

assistant conversations and application screenshots collected on smartphones. It collects personal records from the original content and then extracts relationships between entities. Because graph entities remain linked to their source records, insertions, deletions, and replacements can be applied to the corresponding nodes. The relations also allow queries involving several facts to combine relevant memories along graph paths. A personal graph can instead focus on user knowledge that tasks repeatedly require, without absorbing complete interaction trajectories. In personalized household tasks, a user-profile memory module [75] retains the original trajectories and constructs a separate user profile graph. This graph stores object meanings and routine order separately from the original interaction trajectories. It links each user to the relevant knowledge and to the objects, locations, and routine steps involved. Information from different memories can thereby be combined, and local connections can be revised as the underlying knowledge changes. Other personal-graph designs include dedicated user profiles in MOBIMEM [92], user-editable smartphone memories in RUVA [35], and temporal health memories in ECHO [73].

The cost of graph access also depends on the representations used for retrieval and reasoning. MobileKGQA [1] replaces high-dimensional embeddings with compact learned binary codes and adapts a lightweight graph reasoner using questions synthesized from updated graph paths. This design allows graph changes to be incorporated without fine-tuning the full language model.

3.4.2 Lightweight Structured Memory. Full entity–relation extraction can increase the cost of constructing and updating a graph. Another line of work uses selected memory entries directly as nodes and organizes them through lightweight hierarchies or similarity links. Mnemosyne [64] treats each interaction summary as a candidate entry. It first removes content with little long-term value, then checks whether a new summary duplicates an existing record. Summaries that pass these filters are added to the graph and linked to existing nodes through embedding similarity and keyword overlap. Repeated information does not automatically create an additional graph node. It increases the likelihood that an earlier record will be retained and recalled. The resulting units remain complete interaction summaries, whereas the edges mainly represent associative strength between entries rather than typed factual relations. Lightweight structure can also be layered over a conventional local database. SuperLocalMemory [13] stores memories in SQLite and organizes projects or topics through parent–child paths. It then uses keywords and textual similarity to add connections and form multilevel clusters. In this structure, similarity edges indicate that entries are related without assigning a specific semantic type to the relation. Its basic implementation constructs pairwise similarities within a predefined limit on graph size.

3.4.3 Multi-Entity Graphs. Continuous first-person records introduce new entities and capture changes in both entity states and relations among entities. Graph-structured memory can record when an event occurred, how long a relation remained valid, and how an entity changed afterward. EgoSelf [164] first segments continuous video into events in which the wearer interacts with a person or object. It then merges people and objects that recur across events into persistent entities. Events are connected through temporal and causal relations or through participation in the same activity. Participants and objects are linked to the corresponding events. This allows interactions recorded across multiple days to be regrouped around the same entity.

To answer when a relation was valid, temporal information can be stored directly on the edge. EGAgent [121] extracts relations among people, objects, and places from visual descriptions, transcripts, and location text. It stores each relation as an incrementally added tuple in SQLite. Each tuple records the relation, its start and end times, and its source text. To track changes in the same entity across days, EgoGraph [142] stores paired timestamps and descriptions on nodes for people, places, objects, and events. It also stores paired timestamps and descriptions on the

relations among these nodes. When an entity reappears, the system merges it with the existing node and appends its new state.

3.5 Parametric Memory

The memory forms discussed above retain past information as units that can be located or invoked at inference time. Parametric memory takes a different route by incorporating selected facts or patterns derived from experience into model parameters. The retained information is used directly in subsequent prediction or generation rather than retrieved as explicit memory items. For on-device agents, this form can reduce the need to repeatedly add memory content to the model context.

3.5.1 Personalized Parametric Memory. When a device serves the same user over time, recurring behavioral patterns can be captured through personalized training rather than supplied as complete histories before every prediction. PITuning [48] retains such patterns across the parameters of an entire lightweight predictor. It first trains a global intent predictor on population-level behavioral data and distills the model into a lightweight predictor suitable for on-device deployment. On the device, it mitigates bias inherited from population-level training against the user’s long-tailed intents. It then tunes the predictor further on personal history to produce user-specific prediction parameters.

A more lightweight route confines personalization to auxiliary parameters. Low-rank adaptation (LoRA [55]) freezes the base model and trains only a small set of additional parameters, allowing user-specific changes to be stored as a separate module. Cloud-assisted adaptation includes user-data augmentation in CDCDA-PLM [210] and joint selection of interactions and trainable layers in EcoTune [26]. In a federated setting, BitLoRA [136] forms adapters through local client training and parameter aggregation. It freezes the 1.58-bit base weights, updates low-rank adapters on private client data, and communicates only adapter parameters for aggregation. Quantization reduces the deployment cost of the shared base model, and the adapters retain changes learned from user or task data.

Personalization can also involve an individual fact supplied explicitly by the user. MobiEdit [95] writes such facts into model weights using quantized, forward-only gradient estimation on mobile neural processing units. It optimizes a targeted edit and applies a rank-one weight update, avoiding the backward pass required by conventional editing. Individual edits in its 3B smartphone experiments take tens of minutes, making update scheduling important for interactive use.

These approaches differ in both the information retained and the parameter unit that carries it. PITuning captures recurring behavior across an entire predictor, whereas CDCDA-PLM and BitLoRA retain learned changes in separate parameter increments. MobiEdit instead targets individual facts through localized weight changes. Parametric memory can therefore preserve either regularities across experience or explicitly supplied knowledge for later prediction and generation.

3.5.2 Task-Specific Parametric Memory. On-device agents often need to reuse acquired capabilities across tasks. Yet training and storing a complete model for each task causes computation and storage to grow with the task set. Task-specific parameters instead concentrate the relevant knowledge or behavioral patterns in lightweight modules that can be selected, composed, or invoked as needed. Crayon [12] implements this idea by first training a set of base LoRAs with different task capabilities. It then estimates blending weights from a small customization set and deploys the resulting LoRA with an on-device language model. Because the customized LoRA is composed from existing adapters, the device does not need to retrain a model for each new task. Its parameters encode the capability required by the target task rather than an operation sequence that can be replayed step by step.

Task capabilities can also be separated within the model through isolated parameter pathways. BiMemNet [80] studies continual learning for visual object detection on an NVIDIA Jetson edge platform. It first compresses multiscale visual features into low-dimensional representations to reduce storage requirements for training data. It then activates an independent parameter pathway for each new task, limiting interference with previously learned tasks. The first mechanism stores compressed training features, whereas the second retains each task’s learned outcome in a dedicated parameter pathway.

Although Crayon’s customized LoRA and BiMemNet’s task-specific pathways differ in form, both use model parameters to carry reusable task capabilities. This shared role across different parameter structures clarifies how the category should be drawn. The relevant criterion is what the parameters retain, not the particular type of parameter module used. Recent agent research provides further examples of this parameter-resident view. TMEM [122] distills supervision from accumulated interaction history into online-updated LoRA weights, allowing the resulting fast parametric memory to directly alter subsequent agent decisions. PEAM [51] selectively consolidates successful and failure–correction trajectories into per-category isolated LoRA adapters, turning selected embodied experience into reusable parametric skills. Related work targeting compact memory agents for resource-constrained deployment, such as DuoMem [54], combines explicit procedural memories with parameter-space distillation, using successful teacher trajectories to transfer behavioral patterns into lightweight LoRA adapters. MemLoRA [16] makes this boundary explicit. Like Crayon, it uses LoRA. However, its adapters are trained for memory extraction, updating, and memory-conditioned generation, whereas the dialogue memories themselves remain outside the model parameters. These adapters learn how to operate on external memory rather than store the user’s experiences. Parameters that directly retain reusable task capabilities constitute parametric memory. When parameters only operate on external records, those records still belong to textual, vector, or graph-structured memory.

3.5.3 Merging Parameter Units. Maintaining separate parameters for every task can reduce interference, but the number of stored parameter units grows with the supported task set. For adapter sets known before deployment, HydraOpt [24] exploits common structure among low-rank adapters and reorganizes them into shared components and task-specific increments. This consolidation is performed on a server before the resulting adapters are deployed to mobile devices. When adapters continue to arrive after deployment, the problem shifts from one-off consolidation to continual capacity management. K-Merge [128] stores each incoming adapter independently while capacity remains. Once the fixed-capacity pool is full, it merges the new adapter with its most similar resident adapter using weights derived from the merge history. HydraOpt and K-Merge both control the number of independently stored parameter units by changing the extent of sharing among task adapters. They frame the central trade-off in on-device storage for multiple tasks as how many parameter units should remain independent under a given storage budget.

3.6 Choosing and Combining Memory Forms

The five forms reviewed above are not mutually exclusive system designs. When an on-device agent retains a personal fact, the content may need to remain inspectable and directly editable. Reusing task experience places greater emphasis on recognizing a similar situation and recovering the relevant procedure. Repeated context and stable patterns accumulated across interactions may instead be reused through cached states or model parameters. A system therefore need not rely on a single memory form. Rather, each memory object should be retained in a form suited to its access pattern and update requirements. Table 3-2 compares the main strengths, limitations, and applicable scenarios of the five memory forms.

Table 3-2. Strengths, limitations, and applicable scenarios of memory forms

Memory Form	Main Strength	Main Limitation	Applicable Scenarios
Textual	Direct inspection and editing	Context cost of retrieved text	Dialogue assistance, task tracking
Vector	Semantic retrieval or compact encoding	Limited direct interpretability	Interface assistance, multimedia search
KV cache	Reuse of context computation	Model version and context compatibility	Repeated tool calls, long conversations
Graph	Explicit relations for reasoning and action	Cost of maintaining valid relations	Personal knowledge queries, event tracing, GUI execution
Parametric	Reuse of learned or edited knowledge	Cost of selective parameter updates	Personalized prediction, knowledge editing, task adaptation

One way to combine memory forms is to preserve memory content separately from the structure used to locate it. This does not duplicate the same content. Instead, it allows an inspectable and editable record to be reached through different access paths. Procedural experience from mobile interfaces provides a clear example. UI-Mem [176] stores workflows, reusable steps, and experience from failed trajectories as textual templates. It then builds vector indexes over task and step descriptions to locate the relevant templates. The textual templates preserve the experience itself, whereas the vectors provide semantic access. When several personal facts must be connected, the access structure can also encode relations among records rather than relying on similarity search alone. EMG-RAG [166] links personal records extracted from smartphone conversations and application screenshots to graph entities. It first uses vectors to narrow the search space and then traverses relational paths to combine relevant records. Viewed in terms of their roles, neither system replaces explicit records with its access structure. Instead, vector indexes and graph relations provide different query paths to records that remain directly revisable.

Another combination strategy assigns different kinds of memory objects to different memory forms. It separates user or task knowledge from the concrete information used during execution. MOBIMEM [92] organizes user facts, preferences, and behavioral patterns in a graph, stores transferable task experience as textual templates, and retains fine-grained action sequences separately. This organization allows user knowledge, task templates, and concrete operations to be revised separately rather than rewritten as a single memory unit. Mobile interfaces present a related need because task structure and interface evidence do not necessarily change together. MAG-NET [141] separates abstract workflows from instances of interface elements, allowing task steps and the evidence used to locate action targets to be maintained independently. In both systems, distinct memory objects remain separately addressable even when they contribute to the same task. HyMem [203] retains event summaries in a vector index and links them to the original dialogue, allowing retrieval to move from a compact summary to its supporting detail.

Combinations of memory forms can also be confused with storage tiering in on-device systems. Both can distribute information across multiple components, but moving the same memory unit between storage locations does not by itself introduce a new form. This distinction is especially important for KV-cache management. KVSswap [195] keeps the full KV cache in local storage, retains a compact relevance representation in memory, and loads selected cache groups on demand. Two different changes are involved. Moving the full cache changes where the same states reside, whereas the compact representation creates a new access unit for assessing and selecting cache groups. By contrast, DynaKV [158] primarily changes how the same KV states are grouped and

laid out across smartphone memory and flash storage. This alters cache access and transfer costs without introducing new memory content.

4 LIFECYCLE MANAGEMENT OF ON-DEVICE AGENT MEMORY

The previous chapter examined what carries memory in on-device agents by organizing prior work around memory forms. The resulting taxonomy provides a structural framework for understanding agent memory. Yet this largely static view does not explain how memory operates and changes once an agent runs continuously on a device. On-device agents continuously accumulate local interaction histories during everyday use. Evidence within these histories may be transient, and its relevance and reliability can change with tasks and user behavior. Memory operations also compete with foreground inference for device resources. Therefore, lifecycle decisions need to be selective in both content and timing. The state changes produced by these decisions constitute the memory lifecycle, whereas lifecycle management coordinates them under local resource constraints.

This chapter organizes prior work around four operations in a lifecycle shaped by device constraints.

- **Memory Formation** (Section 4.1) selects which local observations become persistent memory and creates units with explicit boundaries and provenance. On-device systems balance missed transient evidence against continuous sensing and write costs.
- **Memory Retrieval** (Section 4.2) determines when memory is accessed, which local store or index is searched, and what content enters the reasoning context. It controls how much index state and retrieved evidence are loaded into limited memory.
- **Memory Evolution** (Section 4.3) integrates new experience with existing memory through consolidation and updating. Systems can schedule and prioritize these operations to limit interference with foreground tasks.
- **Memory Forgetting** (Section 4.4) reduces or removes information that should no longer influence the agent. In on-device systems, deletion must also be traced from source records into derived representations and parametric state.

These operations are interdependent and form a recurring cycle during device use. Newly formed memories enter the local store and may be integrated with existing content through consolidation and evolution. Retrieval selects from this changing store according to the current task and the resources available at that moment. Task outcomes and environmental feedback can then create new memories or trigger updates to existing ones. Selective forgetting reshapes what remains accessible and can reduce both stale influence and future maintenance costs. Lifecycle management therefore does more than improve recall. It keeps memory useful and maintainable without overwhelming the resources needed for ongoing tasks. It also allows obsolete or deleted content to stop influencing later behavior. Figure 4-1 visualizes this lifecycle around a shared on-device memory state.

4.1 Memory Formation

An on-device memory lifecycle begins before anything is committed to persistent storage. Events on personal devices can be transient, so continuous capture appears to offer the safest coverage. In practice, continuous sensing, semantic interpretation, and storage would consume resources while retaining private material unrelated to later tasks. Existing systems place admission decisions at different points between sensing and persistent storage.

4.1.1 Write Admission. The earliest admission decision occurs before an expensive modality is activated. At this stage, a low-power signal is used to detect candidate events without interpreting the full scene. SAMoSA [105] first identifies possible activities from motion signals and then records

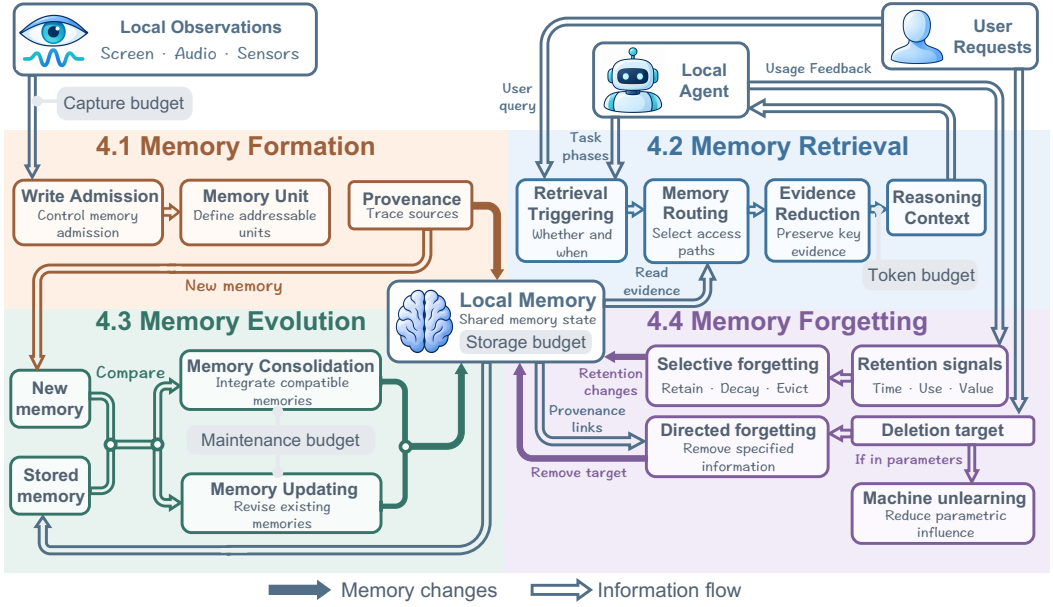


Fig. 4-1. Lifecycle management of on-device agent memory. Filled arrows indicate memory changes, whereas outlined arrows indicate information flow.

sparsely sampled audio. EgoTrigger [112] uses sounds produced by hand–object interactions to trigger image capture on smart glasses.

Once raw observations have been captured, admission can move to the boundary before costly semantic interpretation. Lightweight contextual cues first remove uninformative observations, leaving a smaller set for a more expensive model. FOCAL [190] applies this design to continuous desktop activity. It filters invalid events and uses the active application and window title to assign actions to tasks. Only selected actions are interpreted by a vision-language model and written to the corresponding task record.

Semantic interpretation still produces candidate memories and does not by itself justify permanent retention. A later admission point can use task dependencies, application requirements, or observed feedback to control persistent writes. STAMP [154] provides explicit supervision for when transient information in mobile tasks should be stored and when later steps will need it. EMBER [81] retains verbatim evidence and provenance under a fixed budget when future queries are not yet known. Application-specific filters include predefined preference categories in CarMem [72] and execution and performance checks in CBR-augmented R&D-Agent [138]. More general learned memory managers, including Memory-R1 [180] and AgeMem [191], treat adding, modifying, deleting, or preserving an entry as learnable choices. They optimize these choices from downstream task feedback.

All three control points narrow the information that continues through the formation pipeline, but each controls a different component of cost. Early admission can prevent sensing and semantic processing overhead. However, excluded observations cannot later be recovered. Later admission uses richer evidence, but the upstream cost has already been incurred. The appropriate control point depends on which resources dominate and how costly missed evidence would be for the target task.

4.1.2 Memory Unit. Admission determines whether candidate information continues through the pipeline, but it does not determine the granularity of the resulting memory. Frame-by-frame storage creates many near-duplicates, whereas fixed temporal windows can split a coherent event. Memory formation converts a continuous stream into addressable units whose boundaries remain meaningful for later use.

Existing systems draw these boundaries from either perceptual continuity or task continuity. Venus [189] segments and clusters a video stream into scenes, selects representative frames, and builds a hierarchy that preserves the visual progression of an event. Temporal proximity becomes less reliable when several activities are interleaved. By contrast, FOCAL [190] groups actions that may be separated in time into the same task record, preserving the continuity of interrupted desktop activities.

Unit construction also requires a choice about which observations adequately represent an event. TriPSS [22] clusters candidate frames using perceptual, structural, and semantic information, then selects frames that balance quality, diversity, and redundancy. For textual histories and tool traces, the criterion shifts from visual coverage to evidential support. EMBER [81] keeps source spans that may support future answers and attaches retrieval keys and update metadata to the retained evidence.

Experience can also be condensed into reusable units, including task templates in UI-Mem [176], common workflow stages in MAGNET [141], and compact rules derived from model errors in PKR [151].

Memory units are useful when their boundaries preserve the events, procedures, or regularities on which later reasoning depends. Formation determines both what evidence remains available and how that evidence can be addressed. Retrieval can select within a stored unit, but it can only operate on observations and relations preserved during formation.

4.1.3 Provenance. Many on-device memories are derived through visual description, summarization, or relation extraction. These derived entries are interpretations of an observation rather than the observation itself. Without provenance, neither the user nor the system can identify the supporting interaction or revisit the source when an interpretation proves incorrect.

One design links each derived entry to a source record or a bounded task context. EMG-RAG [166] connects graph nodes to the underlying smartphone records, allowing insertion, replacement, and deletion to target the relevant graph partition. FOCAL [190] expresses provenance through separate task records. Similar-looking desktop actions remain attached to the activity in which they occurred instead of being merged into a global history.

Provenance also needs to preserve how information changes over time. APEX-MEM [11] attaches facts to temporally grounded, append-only events rather than silently overwriting earlier versions. Conflicting versions remain available when a later query requires temporal resolution. SelfMem [184] instead retains the original dialogue as an immutable source. The agent can revise derived memory in a separate workspace.

Provenance is more than descriptive metadata. It provides the linkage required to revise, verify, or remove a derived memory without losing sight of its source. For personal information, this traceability also supports reliable user control. Section 5.4 returns to its role in privacy protection and auditability.

4.2 Memory Retrieval

Once memory has been formed, an on-device agent faces the question of how stored information should re-enter an ongoing task. A complete index may not fit in device memory, and not all

retrieved evidence can fit a small model’s context. Under these constraints, the first retrieval decision is whether and when memory access is warranted.

4.2.1 Retrieval Triggering. A user query is the most common retrieval trigger, but it can control different parts of the access process. MemFlow [27] interprets the request before choosing whether to load stable user information, retrieve a specific fact, or prepare a deeper body of evidence. MobileRAG [111] uses the query to narrow the local index partitions loaded into memory. For questions over long multimodal experience, EGAgent [121] decomposes the query into evidence needs before initiating searches over visual evidence, audio evidence, or relations.

For an acting agent, the initial request is only one possible trigger. Retrieval may instead be tied to the stage of execution. UI-Mem [176] retrieves workflows, subtask experience, and failure records before a mobile interface task so that prior experience can guide exploration. MOBIMEM [92] similarly accesses user information and task experience at task onset. Stored action sequences are invoked only when the current interface satisfies their reuse conditions. EfficientNav [185] revisits retrieval during navigation as the target and explored environment change, loading map content relevant to the current planning stage.

Queries, task phases, and environmental changes provide different signals for conditioned access. A restrictive trigger can leave the agent to act without necessary historical evidence. An overly permissive trigger repeatedly loads indexes, occupies context, and exposes records that the current decision does not need. A useful policy triggers long-term memory when the task depends on it and avoids access when the active state already provides sufficient evidence.

4.2.2 Memory Routing. After retrieval has been triggered, routing determines which memory pathway should handle the request. In systems with several functional stores, the route follows the role that memory is expected to play. MemFlow [27] sends a request to stable user information, targeted factual retrieval, or deeper evidence preparation. MOBIMEM [92] separates user information, reusable task experience, and action records associated with matching interface states. This separation allows personalization, planning, and execution to draw on different stores.

Longitudinal multimodal experience extends routing from functional stores to evidence sources. EGAgent [121] assigns query subproblems to visual search, audio transcripts, or entity relations before combining the filtered results. However, the relevant source cannot always be chosen in one step. REMem [133] allows reasoning to alternate among semantic retrieval, lexical search, and relation exploration under temporal constraints. REMem provides a general multi-cue access pattern rather than a measured on-device implementation, so the cost of iterative tool use depends on the deployment setting.

Routing also has a physical layer when the relevant index or memory partition cannot remain resident in RAM. MobileRAG [111] loads only the index partitions selected for the current query. EfficientNav [185] chooses map groups relevant to the target under a device memory budget and loads their associated cache state. These mechanisms determine which otherwise relevant content is physically available to the model.

Memory routing maps a current need to both an information source and an accessible portion of storage. Functional routing identifies the role of memory, modal routing identifies the evidence type, and physical routing limits what is loaded. On personal devices, these choices remain subject to user, application, and permission boundaries. Semantic relevance indicates what could help, whereas access control determines what may participate in the task.

4.2.3 Post-Retrieval Evidence Reduction. Initial retrieval and routing may still return redundant or weakly relevant candidates. Passing them directly to a local model consumes context and can

distract reasoning. Post-retrieval evidence reduction narrows the candidates used for the current call without changing the underlying long-term store.

For textual evidence, reduction can remove irrelevant spans, reorder records, or select a smaller candidate set. Examples include sentence filtering in MobileRAG [111], priority-based context packing in MemFlow [27], small-model candidate selection [197], and low-confidence rejection in MemX [140].

Candidate evidence may also contain complementary information that should not be filtered independently. One clustering-driven memory compression method [17] first selects personalized documents relevant to the request and converts each document into a compact continuous representation. It then groups similar representations and merges candidates within each group. Collapsing heterogeneous content into one representation degraded performance in its experiments, suggesting that reduction should preserve meaningful distinctions among candidates.

Post-retrieval reduction can consider more than topical relevance. MemGuide [43] reranks retrieved memories according to whether they supply conditions still missing from the current goal. EGAgent [121] reduces evidence separately within each modality before combining it for multimodal reasoning. HiAgent [56] replaces completed subtask histories with compact summaries while retaining a route back to the underlying trajectory. Although HiAgent is not an on-device system, its design shows how context reduction can preserve access to supporting detail.

The goal of post-retrieval reduction is not maximal compression, but the preservation of evidence relevant to the decision within a limited context. Filtering, aggregation, and reranking leave the long-term store unchanged, yet they can still remove decisive information after retrieval has succeeded. Evaluation should therefore report both context savings and the retention of key evidence, allowing reduction failures to be distinguished from retrieval failures.

4.3 Memory Evolution

Task execution and environmental feedback continue to generate evidence after a memory has been formed. If new records are appended without integration, the store becomes fragmented and conflicting information remains unresolved. Memory evolution coordinates new evidence with stored content so that the memory store can continue to support later tasks.

As used here, evolution begins after initial formation. Consolidation integrates information that can remain jointly valid, reducing fragmentation and creating more stable access units. Updating responds when new evidence changes the applicability of an existing fact, procedure, or state. The two operations often rely on the same comparison between new and stored memories, but they serve different purposes. An update may invalidate an old entry to correct the record, whereas Section 4.4 addresses removal intended to reduce an item’s future influence.

4.3.1 Memory Consolidation. Under a tight maintenance budget, consolidation usually begins with a new entry and a small set of nearby memories. The first decision is whether the new content should extend an existing entry or remain separate. UI-Mem [176] retrieves semantically similar experience after extracting knowledge from a mobile interface trajectory. It updates success statistics or timestamps when a match exists and registers a new experience otherwise. MAGNET [141] preserves a new workflow as another version for the same task goal, whereas newly observed interface elements extend the corresponding records.

Dialogue memory systems apply the same local principle to repeated evidence. Mnemosyne [64] filters interaction summaries with little long-term value and links recurring content to existing records. RMM [144] organizes a conversation into entries defined by topic and retrieves a small set of similar memories for each new entry. A language model then decides whether the entry

should be added or merged. In each case, consolidation is restricted to the new entry and a limited neighborhood rather than the full memory store.

However, relevant information is not always concentrated in a few nearest neighbors. A-Mem [179] extends consolidation over a local relation network. After adding a new memory, it creates links to related records and revises their contextual descriptions, keywords, or tags to reflect the new relations. Consolidation can also span multiple interactions through a higher-level summary. Mnemosyne [64] periodically selects a fixed number of representative graph nodes and combines them with the existing long-term user summary. This allows accumulated evidence to revise the summary without reprocessing the entire dialogue history.

Consolidation changes both the size of a memory store and the granularity at which its contents remain accessible. A wider consolidation scope can absorb relations distributed across more entries, but it also requires additional model calls and storage writes. Errors can propagate into more derived content as the scope grows. For on-device systems, consolidation should reduce redundancy within a maintenance budget while preserving distinctions needed for later correction.

4.3.2 Memory Updating. Consolidation coordinates information that can coexist, whereas updating asks whether an existing memory still applies. User circumstances, software interfaces, and task environments all change over time. Updating responds by correcting an entry, preserving successive versions, or marking an earlier state as no longer valid.

When change appears as an explicit factual conflict, the update can occur as soon as new evidence arrives. Mem0 [32] compares a candidate fact with related records and chooses whether to modify, delete, or preserve the existing entry. Zep [119] does not simply overwrite an older relation. It marks the relation as invalid and records its validity and invalidation times, preserving when the fact held and when it changed.

Some changes become visible only when a stored procedure is reused. MOBIMEM [92] checks the current interface before replaying an action. If the state no longer matches, the agent abandons the cached action and returns to planning. It then revises the corresponding record from the new trajectory. ActionEngine [208] likewise uses repeated execution failure to localize an obsolete interface operation. It relocates the target element and repairs the failed step and its associated information instead of rebuilding the entire interface memory.

Other memories become questionable without producing a direct conflict or interface error. Memory-R1 [180] learns whether to add, modify, remove, or preserve entries in dialogue question answering and simulated long-horizon tasks. Its policy is optimized according to whether the resulting memory supports a correct answer. AgeMem [191] jointly optimizes long-term memory operations and management of the active context, allowing task completion, context control, and memory quality to shape the policy. These systems use downstream outcomes when the appropriate update cannot be specified in advance.

Explicit conflict, execution mismatch, and task outcome provide progressively less direct evidence about what should change. Updating uses these signals to keep past experience aligned with a changing user, task, or environment. Its central challenge is deciding whether an observed mismatch reflects a persistent change or a temporary exception. A premature update can turn an isolated event into durable memory, whereas a delayed update leaves obsolete experience available for reuse.

Consolidation and updating also share a scheduling problem on resource-constrained devices. The need for maintenance can be detected during interaction without placing every model call, memory read, and storage write on the foreground path. LightMem [46] moves heavier maintenance out of online interaction and processes accumulated work offline. MOBIMEM [92] uses priority queues so that stale actions and experience likely to affect near-term execution are handled before

less urgent profile changes. HUOZIIME [126] applies this separation to an Android input method, retrieving personal facts during text completion and extracting new facts after a session ends. New input preempts background processing, giving interaction priority over memory maintenance. Local failures that block the current task may require immediate repair, whereas broader merges and reorganization can be deferred until resources are available.

4.4 Memory Forgetting

Memory evolution keeps useful experience aligned with later tasks, but an accurate memory does not necessarily deserve indefinite residence or influence. As the store grows, an on-device system must adjust what remains accessible and remove specified information when required. For explicit memory, selective forgetting lets the system choose entries for decay or eviction from runtime signals. Directed forgetting begins with a specified target and propagates its removal through related memory artifacts. When the same information has entered model parameters, machine unlearning is required to reduce its learned influence.

4.4.1 Selective Forgetting. Selective forgetting uses signals generated during the lifetime of a memory to decide which entries remain accessible, which gradually lose priority, and which are removed from storage. Existing approaches mainly rely on elapsed time, usage state, and task value.

Forgetting based on elapsed time treats age as the primary retention signal. Mnemosyne [64] assigns graph nodes a selection probability that decays over time, reducing the chance that unrefreshed memories participate in later tasks. Repeated or newly used content can be refreshed, whereas nodes that remain infrequently accessed are pruned only when the graph approaches its capacity limit. Older memories lose accessibility before they are removed from the graph. Researchers are also exploring gradual reductions in memory fidelity, including age-dependent visual compression in ScrapMem [25] and retention-guided embedding precision in SuperLocalMemory V3.3 [14].

Forgetting based on use considers whether a memory continues to participate in interaction. MemoryOS [65] combines access frequency, interaction length, and time since last access into a state score. When mid-term storage reaches capacity, low-priority segments are evicted, whereas content that remains active can move into long-term personal memory. Retention is thereby tied to the subsequent use of an entry rather than its creation time alone.

Forgetting based on value relates task benefit to the device resources consumed by a memory. CURATOR [168] combines the expected benefit and possible harm of an experience with its byte cost. It evicts low-value experience under a local budget. On heterogeneous Jetson robots, CURATOR reduces memory use, energy consumption, and uplink traffic while maintaining overall task accuracy. Memory Worth (MW) [135] scores each memory according to how often its retrieval coincides with successful and failed outcomes. The score can lower access priority or prompt an entry to be reviewed or disabled. MemArchitect [74] similarly updates utility and stability after a memory contributes to an answer, allowing subsequent decay and background cleanup to change its accessibility. In these methods, retention value can change with observed use and resource cost rather than remaining fixed at write time.

Selective forgetting is better understood as ongoing control over retention than as a single deletion event. Time supplies a basic decay signal, usage state indicates whether an entry remains active, and task value incorporates observed benefit and device cost. These signals can jointly determine which memories remain readily accessible and which gradually leave routine access or local storage.

4.4.2 Directed Forgetting. Unlike selective forgetting, directed forgetting starts from an already specified deletion target. Once a request has been accepted, the system must locate the source

record and propagate the request to summaries, indexes, or other derived artifacts. Section 5.4 discusses authorization, user control, and the privacy significance of deletion requests. The present section focuses on how an accepted target is removed from the memory system.

A study of deployment-time memorization [30] examines deletion across source records and derived summaries. Removing only the source left the target recoverable from the summary layer in about 20% of its test cases. Regenerating, deleting, or masking the corresponding summaries reduced the tested textual residue to zero. The provenance links discussed in Section 4.1.3 become deletion paths in this setting because they connect a source to its derived content. The experiment directly establishes this relation for source text and summaries. However, extending the same linkage to vector indexes, caches, and other local replicas remains a system requirement rather than a result demonstrated by that study.

For agents that continue to perceive and act, directed forgetting may also concern how past experience shapes future behavior. A natural-language-based agent-unlearning framework [187] defines targets at the state, trajectory, and environment levels. It removes the relevant experience and converts natural-language requests into constraints on subsequent behavior. In GridWorld, ALFWorld, and Gazebo simulations, the agent avoids the specified states or trajectories while continuing to complete unaffected tasks.

Directed forgetting must remove the target from source records and derived content and close the access paths through which it could influence later tasks. Deleting one copy is insufficient when another representation can still return the information to the agent. Completion should be judged by whether the system can still retrieve the target or use it to guide action, not by whether one record was deleted.

4.4.3 Machine Unlearning. Selective and directed forgetting operate on explicit memories that can be located and manipulated. When training or continual adaptation has absorbed information into model parameters, deleting external records does not reverse the model’s learned response. Machine unlearning addresses this parametric influence while seeking to preserve capabilities unrelated to the target.

Machine unlearning can use either exact retraining or an approximate parameter update. CAUSE [175] partitions training data by user and retains compressed intermediate models on the device. When a deletion request arrives, only the affected shard needs to be retrained. Measurements on a Jetson Orin Nano cover retraining time, memory use, and energy, showing how retained intermediate states can shorten exact unlearning on a constrained device. CAUSE provides device-level evidence for exact unlearning, although its experiments concern conventional learning workloads rather than long-lived agent memory.

Approximate methods reduce the cost of changing parameters. ZO-GA [193] replaces back-propagation with zeroth-order gradient ascent using forward evaluations, lowering peak memory during the update. Orthogonal Entropy Unlearning (OEU) [199] supports unlearning in quantized models. It increases predictive uncertainty on the target data while adjusting the update direction to preserve behavior on retained data.

Execution can begin only after a user-facing request has been translated into an operational unlearning target. Many methods assume that the forget set is already available. However, an agent may receive only an entity, fact, or natural-language request. MAGE [79] probes related information in the model and synthesizes supervision for unlearning without access to the original training corpus. RePAIR [117] takes the interaction itself as the starting point. It detects an unlearning intent, extracts the target, and applies a closed-form activation update that changes the relevant intermediate representation. A low-rank variant reduces the cost of each repair and achieves up to roughly threefold acceleration in experiments with Llama-3-8B. The pipeline is designed for

interactive requests, but evidence for on-device use currently comes from complexity and runtime rather than full device integration.

Local parameter modification also raises the question of how an external verifier can confirm that the prescribed process was executed. ZK-APEX [98] encodes approximate unlearning as verifiable computation and generates a zero-knowledge proof while updating a personalized model. A verifier can confirm protocol-compliant execution without accessing the device's private data. The study demonstrates block-level proof generation on an iPhone 14 Pro Max, although producing the complete proof remains time-consuming. This execution evidence provides a technical basis for auditing unlearning operations, a role revisited as part of the privacy guardrails in Section 5.4.

Machine unlearning covers parametric influence that deletion from explicit memory cannot reach, but it does not replace selective or directed forgetting. When the same information appears in both external records and personalized parameters, the target scope must remain consistent across the two layers. External access paths and influence within the model then require separate, coordinated removal.

5 PERSONALIZATION THROUGH ON-DEVICE AGENT MEMORY

Personalization is where on-device memory becomes more than a local substitute for cloud-hosted conversation history. A general-purpose model can act on information supplied in the current exchange, whereas a personal device can observe recurring patterns across days, applications, and physical settings. Phones capture cross-application actions, smart glasses accumulate first-person experience, and wearables record physiological and environmental context. These longitudinal signals provide evidence about a particular user that is unavailable to a general model [96, 157, 183].

The value of long-term records for personalization depends on how well an agent interprets them in the context of a particular task. The agent must infer preferences and check their stability against recurring behavior and later feedback. It must also link private expressions to the user's people, objects, and experiences, and decide whether a recurring need warrants proactive assistance. The same evidence is sensitive, so its retention, access, and disclosure must remain controllable and auditable. This chapter examines user preferences (Section 5.1), personal reference resolution (Section 5.2), and proactive assistance (Section 5.3). The privacy safeguards discussed in Section 5.4 apply across these uses, as summarized in Figure 5-1.

5.1 User Preferences

User preferences are relatively stable tendencies among alternative contents, outcomes, or ways of completing a task. Explicit settings provide high-confidence evidence and are easy to explain, but they cover only preferences that users can anticipate and are willing to configure. Many preferences instead surface through clicks, dwell time, navigation paths, query reformulations, and repeated choices, a premise established in early work on implicit feedback [63, 67]. Personal devices make this evidence especially valuable because they observe small decisions repeatedly in the settings where those decisions occur. Recent mobile agents consequently use long-term interaction histories to recover patterns that users have never stated as rules [96, 157].

User behavior offers clues to preference, but the meaning of a choice depends on the circumstances in which it was made. A single choice may reflect a temporary compromise, and a skipped option may be caused by circumstances rather than dislike. Reliable inference therefore depends on whether a pattern recurs under comparable conditions and whether later feedback supports the same interpretation. Me-Agent [157] learns user-specific interaction tendencies from mobile histories, whereas HIM-Agent [96] groups related intents and action traces to identify choices that persist across tasks. PAHF [83] approaches the same problem through interaction. When an instruction is ambiguous and no relevant preference can be recovered, it asks for clarification to inform

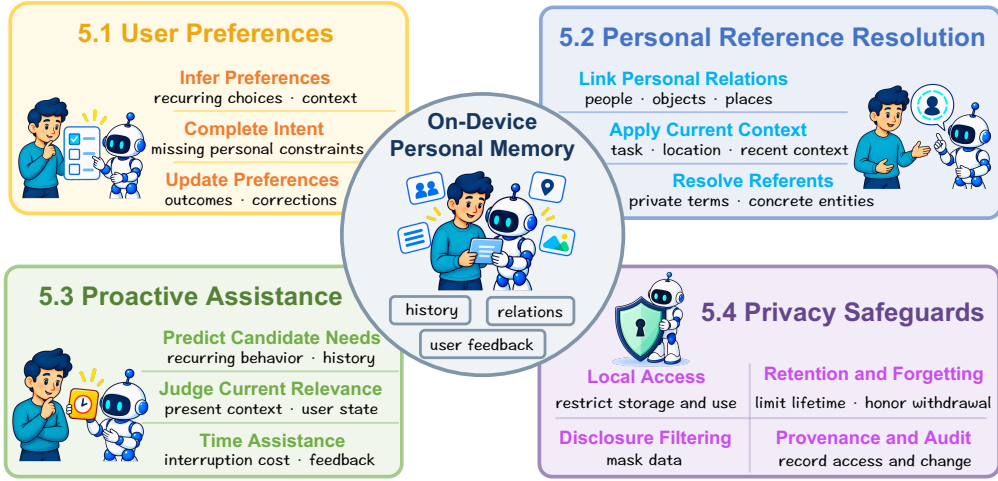


Fig. 5-1. Roles of personal memory in on-device personalization and the privacy safeguards that constrain them. The figure shows how user-specific information supports preference learning, personal reference resolution, and proactive assistance, while privacy safeguards constrain the entire personalization system.

subsequent preference judgments. Across these approaches, behavioral histories, action traces, and user clarification provide complementary evidence for distinguishing persistent preferences from isolated choices.

The practical value of a learned preference appears when the current instruction leaves a personal condition unstated. Requests such as “play something I usually listen to” or “order it the way I normally do” are linguistically clear. Public knowledge alone cannot supply the missing personal details. PerPilot [159] first identifies which part of a mobile task requires personal information. When suitable records exist, it retrieves the relevant information. Missing evidence can instead prompt exploration of likely applications, with the result retained for later tasks. HIM-Agent [96] instead derives recurring fine-grained choices from prior GUI interactions and uses them to complete underspecified requests. MemAura [89] extends this use of history to smart environments. It organizes device logs into temporal event graphs and user-specific behavior statistics, combining them with periodic predictions and a compact preference queue. This structured context helps small language models interpret commands in light of the user’s routines. Other smart-space frameworks, such as UserCentrix [124], reuse recent preferences and previously evaluated plans to condition new requests. Across these systems, memory supplies personal constraints left unstated in the current instruction, while the active task determines the intent to be fulfilled.

A preference inferred once cannot be treated as a permanent conclusion. Changes in work, routine, or life stage can make an earlier choice obsolete even when the agent remains confident in it. PAHF [83] addresses this problem by placing preference learning on both sides of an action. Before acting, the agent can clarify an ambiguous request when memory offers no relevant answer. After acting, corrective feedback updates beliefs that produced an unsuitable result. Pre-action clarification alone may miss a stale preference that the agent confidently treats as current. PERMA [90] evaluates this temporal problem with event sequences in which preferences emerge, receive additional evidence, or conflict with earlier observations. Current memory methods benefit from the accumulated history but deteriorate when preferences change across contexts or when old and new evidence disagree.

Preference learning on personal devices is therefore a continuing loop. Longitudinal behavior supports an initial inference, the inferred preference helps interpret the present task, and the outcome provides new evidence. The objective is to preserve tendencies supported by recent evidence while allowing later behavior or explicit user feedback to update them.

5.2 Personal Reference Resolution

Preferences explain what a user tends to choose, but many personal instructions also depend on what particular entities mean in that user’s world. Public knowledge can define a supervisor, a home, or a favorite object in general terms. It cannot identify which person, place, or possession the speaker intends. PersONAL [214] tests whether embodied agents can use household ownership relations when carrying out personalized tasks. MEMENTO [75] extends the challenge to indirect descriptions whose referents depend on prior interactions and the personal significance of objects. In an auxiliary experiment that replaced explicit object references with indirect ones, both evaluated models suffered lower task success rates. The decline highlights the difficulty of connecting a new expression to the user’s private relations, even when relevant records have already been stored.

Those relations are rarely stated in a single observation. They are distributed across encounters with the same people, places, and objects, sometimes months apart. EgoSelf [164] extracts events from long-term first-person streams and connects experiences through recurring entities. These links offer relational clues, but the evaluation of personalized egocentric question answering does not isolate the resolution of new indirect expressions. An isolated record may show that an object was present, whereas linked episodes can reveal who owns it, why it matters, and how its role has changed.

Personal relations also have contextual boundaries. SpeechLess [71] uses prior interactions associated with locations and activities to interpret partial or even absent utterances. Its evaluation found that partial and zero-utterance interaction reduced spoken effort. Successful question-answering rates were 86.7% and 83.3%, compared with 95.4% for full utterances. Context can recover much of the missing expression, although the remaining accuracy gap reflects the cost of relying on implicit evidence. The challenge becomes more severe when the relevant context spans years and multiple data sources. ATM-Bench [102] constructs queries over approximately four years of personal multimodal records, including cases in which later evidence supplements or conflicts with earlier information. Accuracy remains below 20% on its more difficult cross-time and cross-source subset. These results show a gap between storing experiences and resolving a current personal reference accurately.

Personal reference resolution consequently depends on both relational and situational evidence. Relations narrow a private term to the entities it could denote, whereas the current task and recent context determine which candidate is intended now. A request such as “send this file to my boss” depends on both forms of evidence. Personal relations identify the recipient, while the active task context determines which file is intended. Successful execution requires the agent to recover both meanings and bind them to the present instruction.

5.3 Proactive Assistance

Preference learning and personal reference resolution shape the response to an explicit request. Proactive assistance draws on the user’s history to anticipate needs before a request is made. It then considers whether the present situation warrants intervention and whether the user would welcome it. On-device agents are well placed to make these judgments because the same devices that record recurring behavior can also observe the conditions under which it reappears.

The first problem is to distinguish a predictable need from mere repetition. HIM-Agent [96] aggregates histories with similar intents and action trajectories, then considers how often and

under what conditions a pattern recurs. Patterns tied to a current request support preference completion, whereas sufficiently regular patterns can motivate a proactive suggestion when the corresponding state returns. PITuning [48] combines an individual’s event history with current context for on-device intent prediction, adapting population-level knowledge to forecast likely future activities. HIM-Agent organizes recurring action patterns, whereas PITuning predicts future activities. Both use long-term regularity, but the timing of an actual intervention requires separate evaluation.

A candidate need inferred from history still has to be grounded in what is happening now. ContextAgent [182] extracts sensory context from wearables and combines it with information derived from historical data. It uses this combined evidence to decide whether proactive service is necessary and which tool should be invoked. ProAgent [181] extends this setting to continuous everyday use. It activates richer perception only when inexpensive contextual cues indicate that more evidence is needed, then reasons over the resulting context together with user preferences. Feedback on an intervention is associated with the situation in which it occurred, allowing later decisions in similar circumstances to be adjusted. Personal history thus gives current observations a user-specific interpretation, whereas current observations prevent a recurring pattern from being acted on outside the circumstances that make it relevant.

Relevance does not settle the timing of an intervention. H-MAPS [108] uses pauses and revisits during desktop reading to trigger personalized literature searches. Its locally maintained memories of recent content, the reading session, and the user profile supply context to a cloud model that formulates search questions. ProMemAssist [114] instead models changes in a user’s working memory from multimodal signals collected by smart glasses. It balances the expected value of assistance against the cost of interruption. In a controlled study with 12 participants, it delivered 130 messages compared with 332 from a prompt-based LLM baseline. Positive responses accounted for 24.6% of its messages and 9.34% of the baseline’s messages. These designs connect memory-based assistance to current reading behavior or cognitive demand, making timing part of the decision to intervene.

Even an apparently well-timed intervention may not be acceptable to every user. Across seven workshops on proactive smart glasses scenarios, participants formulated triggering and blocking rules at different levels of abstraction. Their judgments also reflected different attitudes toward autonomy, practical benefit, and emotional impact [45]. These findings suggest that a personal agent needs to learn which forms of help a user is likely to need and when that user considers intervention appropriate. Reactions to earlier assistance, including acceptance, dismissal, and correction, become evidence for calibrating both future predictions and the agent’s willingness to interrupt.

5.4 Privacy Safeguards

Keeping personal memory on a device reduces exposure, but it does not make the full personalization pipeline private. External model calls can expose sensitive memories [31, 171] or locally stored browser histories [148]. GUI screenshots can also reveal information unrelated to the task [205]. As an agent learns more about its user, failures in retention, access control, or disclosure become increasingly consequential. Existing safeguards can be organized by where they intervene in the flow of personal memory. They begin with storage and access, continue through retention and forgetting, and end with filtering before information crosses a controlled boundary.

The first safeguard keeps personal data local and restricts who can use it. SuperLocalMemory [13] stores personal records and behavioral learning data locally. Per-agent provenance and history-based trust scores distinguish contributions from different agents. OpenJarvis [123] shows that model inference, agent execution, tools, and memory can run together on a personal device. Cloud models may assist the system configuration search, but the accepted configuration runs on device

at inference time. When a personal archive exceeds local capacity, Opal [66] confines sensitive computation to a trusted environment and conceals access patterns to external storage. This design makes it harder for the service provider to infer a user’s interests from retrieval locations and frequency. The protection offered by local storage therefore depends on where data reside and which components can observe their use.

A second safeguard limits how long personal information remains usable. MaRS [5] incorporates sensitivity into retention decisions, assigning different lifetimes to memories with different privacy risks. For user-initiated withdrawal, the scope of a forgetting request must be clear without requiring the user to enumerate every affected record. A natural-language-based agent-unlearning framework [187] accepts high-level requests and converts them into operational instructions that regulate an agent’s later behavior. MAGE [79] requires only a limited anchor for the target and constructs supervision for model unlearning without asking the user to supply the original forget corpus. MAGE addresses parametric model unlearning, whereas the natural-language framework constrains later agent behavior without modifying the language model. Removing an on-device memory record remains a separate task. These approaches suggest that users should be able to scope withdrawal requests without reconstructing all the original data behind a memory.

Accepting a deletion request does not ensure that every representation has been erased. ORISA [178] shows that conversation tokens and KV states can remain recoverable after chat deletion in mobile LLM applications. Given process-memory and storage images, it reconstructs residual context on another machine. Deletion controls therefore need to reach runtime states as well as visible conversation records.

The final safeguard filters personal information before it leaves the controlled environment. AMP [171] detects sensitive spans on device, replaces them with protected placeholders for external processing, and restores the corresponding values after the result returns. MemPrivacy [31] similarly substitutes type-aware placeholders, preserving enough semantic structure for the cloud model to complete the task without exposing the original values. MaskClaw [205] applies this decision to visual interface data. Before a screenshot is transmitted, it can allow the image, mask selected regions, or ask the user to decide. By filtering the context sent to external models, these methods reduce the disclosure of personal information. The extent of protection nevertheless depends on the filtering rules and task configuration.

Privacy safeguards must also make memory operations inspectable. SuperLocalMemory 4.0 [15] links each canonical write to required updates of derived indexes and records whether these operations complete, fail, or remain incomplete. Its deletion receipts report removal from the local components covered by those checks. Mi-Memory [38] extends auditability to memory strategy changes by recording source evidence, stage-level diagnostics, versioned strategies, and acceptance or rollback decisions. Its LiteMem component maintains provenance and revision history in a local Markdown store with optional Git versioning. MaskClaw [205] subjects privacy policy updates derived from feedback to rule checks and sandbox validation before they are accepted. These records help users distinguish a requested change from a completed operation. User-facing controls can then expose what was changed, which derived records were affected, and whether further work remains.

6 ON-DEVICE AGENT MEMORY ACROSS PLATFORMS

Chapters 3–5 described on-device agent memory through its representations, lifecycle, and role in personalization. In working systems, device form factors also shape how memory is formed and used. On phones and personal computers, relevant state may disappear as users move among pages, windows, and applications. Head-worn systems capture first-person or spatial context, while personal sensing systems derive evidence from intermittent signals. Embodied systems couple

stored environmental state and reusable experience to action. These settings differ in available observations, interaction timescales, and the role memory plays during execution. Accordingly, this chapter reviews deployment evidence from personal computing systems (Section 6.1), head-worn systems (Section 6.2), personal sensing systems (Section 6.3), and embodied systems (Section 6.4).

Assessing on-device deployment requires tracing where memory formation, updating, retrieval, and memory-conditioned reasoning actually run. A phone, smart glasses, or a robot may collect observations while memory processing takes place on a companion computer or in the cloud. Within each device context, we therefore examine what has been demonstrated on the target hardware and which resource measurements are available. Deployment evidence concerns the evaluated system or component at runtime. Table 6-1 summarizes these findings for the representative methods discussed below.

Table 6-1. Platforms, deployment evidence, and resource measurements of representative methods

Method	Platform / deployment	Resource measurements			
		<i>R</i>	<i>S</i>	<i>T</i>	<i>E</i>
Personal computing					
Stamp-GUI [154]	Smartphone <i>No local evidence</i>	✗	✗	✗	✗
ATMem [84]	Smartphone <i>No local evidence</i>	✗	✗	✗	✗
TSR [207]	Smartphone <i>No local evidence</i>	✗	✗	✗	✗
UI-Mem [176]	Smartphone <i>No local evidence</i>	✗	✗	✗	✗
MOBIMEM [92]	Smartphone <i>Partial local</i>	✗	✓	✓	✗
PalmClaw [20]	Smartphone <i>Partial local</i>	✗	✗	✓	✗
MAGNET [141]	Smartphone <i>No local evidence</i>	✗	✗	✗	✗
MUSE [202]	Smartphone <i>Full local</i>	✗	✗	✓	✓
MobileRAG [111]	Smartphone <i>Full local</i>	✓	✗	✓	✓
DynaKV [158]	Smartphone <i>Full local</i>	✓	✓	✓	✓
FOCAL [190]	Laptop or desktop <i>Full local</i>	✗	✗	✗	✗
Agent-X [34]	Mac mini <i>Full local</i>	✓	✓	✓	✗
persistent 4-bit KV caches [132]	Mac <i>Full local</i>	✗	✗	✓	✗
LEANN [163]	RTX 4090 workstation <i>Full local</i>	✗	✓	✓	✗
MemLoRA [16]	Personal computer <i>Feasibility only</i>	✗	✗	✓	✗
ActionEngine [208]	Desktop or web GUI <i>No local evidence</i>	✗	✗	✓	✗

R: runtime memory; *S*: persistent storage; *T*: response/task time; *E*: energy or power.

✓: measured (including partial or off-device results); ✗: estimate only or not reported.

Table 6-1 (continued)

Method	Platform / deployment	Resource measurements			
		<i>R</i>	<i>S</i>	<i>T</i>	<i>E</i>
PUBG Ally [115]	Consumer PC <i>Full local</i>	✓	✗	✓	✗
fixed-persona dialogue system [19]	Consumer PC <i>Full local</i>	✓	✓	✓	✗
Head-worn systems					
EgoTrigger [112]	Smart glasses <i>Partial local</i>	✗	✓	✗	✓
ColorTrigger [21]	Wearable or edge camera <i>No local evidence</i>	✗	✗	✓	✗
EgoLife [183]	Smart glasses <i>No local evidence</i>	✗	✗	✗	✗
EgoGraph [142]	Egocentric recorder <i>No local evidence</i>	✗	✗	✗	✗
ProMemAssist [114]	Smart glasses <i>Partial local</i>	✗	✗	✗	✗
personalized proactive assistance study [45]	Smart glasses <i>No local evidence</i>	✗	✗	✗	✗
intention-aware semantic communication [62]	Smart glasses <i>No local evidence</i>	✗	✗	✗	✗
SpeechLess [71]	Quest 3 <i>Partial local</i>	✗	✗	✓	✗
PITAR [60]	XR headset <i>Partial local</i>	✗	✗	✓	✗
ClickAIXR [68]	Magic Leap 2 <i>Full local</i>	✗	✗	✓	✗
Personal sensing					
SAMoSA [105]	Smartwatch <i>Partial local</i>	✗	✗	✓	✓
PrISM-Observer [7]	Apple Watch <i>Full local</i>	✗	✗	✓	✗
PHIA [103]	Wearable data platform <i>No local evidence</i>	✗	✗	✗	✗
ECG-Agent [33]	Smartphone or wearable <i>Feasibility only</i>	✗	✗	✗	✗
Affective Agent [106]	Raspberry Pi CM5 <i>Partial local</i>	✓	✓	✓	✗
MemPal [100]	Neck-worn smartphone <i>Partial local</i>	✗	✗	✓	✗
Embodied systems					
VLONS [129]	Jetson robots <i>Full local</i>	✗	✗	✓	✗
EfficientNav [185]	Jetson (Habitat) <i>Full local</i>	✗	✗	✓	✗
STaR [192]	Husky robot <i>Partial local</i>	✓	✓	✓	✗
MemCtrl [42]	Robot simulators <i>No local evidence</i>	✗	✗	✗	✗

R: runtime memory; *S*: persistent storage; *T*: response/task time; *E*: energy or power.

✓: measured (including partial or off-device results); ✗: estimate only or not reported.

Table 6-1 (continued)

Method	Platform / deployment	Resource measurements			
		<i>R</i>	<i>S</i>	<i>T</i>	<i>E</i>
Memo [52]	Robot simulators <i>No local evidence</i>	✓	✗	✓	✗
Voyager [150]	Minecraft <i>No local evidence</i>	✗	✗	✗	✗
user profile memory module [75]	Simulated Spot robot <i>No local evidence</i>	✗	✗	✗	✗

R: runtime memory; *S*: persistent storage; *T*: response/task time; *E*: energy or power.

✓: measured (including partial or off-device results); ✗: estimate only or not reported.

6.1 Personal Computing Systems

Agents on personal computing systems interact with users primarily through graphical user interfaces, applications, and local files. On smartphones, interface state often disappears after a page transition or application switch, and limited local resources constrain how much history can be retained and processed. Laptops and desktops can host stronger local models but accumulate longer activity histories with less distinct task boundaries. Although both device classes center on interface interaction, they present memory systems with different operating conditions.

6.1.1 Smartphones. Evidence displayed on a mobile interface may disappear after a page transition. Yet later actions may still depend on a verification code, a copied value, or a completed step. To maintain continuity across interfaces, one line of work represents memory as an explicit task state that can be updated over time. This representation avoids repeated reconstruction from the full screenshot history. STAMP [154] uses controlled mobile tasks to annotate which transient information should be retained and when subsequent steps will depend on it. ATMem [84] organizes retained task information by role, dependency, and completion status, whereas TSR [207] derives persistent progress from changes between consecutive interface states. Making task dependencies explicit reduces the burden of recovering progress across application boundaries. As interaction histories grow, smartphone agents also need to preserve operational experience that can be reused in later tasks. The emphasis then shifts from tracking immediate progress to extracting relatively stable task structures from successful and failed trajectories. UI-Mem [176] abstracts successful trajectories into reusable workflows and action steps while retaining failure patterns to guide subsequent exploration. Mobile frameworks also organize persistent context into separate user, task, and action stores in MOBIMEM [92] or shared personal memory and session summaries in PalmClaw [20]. MAGNET [141] distinguishes relatively stable interface semantics from procedures that may change across software versions.

These approaches address what should be retained during mobile tasks. A systems-oriented line of work asks how memory can be stored and accessed under hardware constraints. MUSE [202] co-designs vector insertion, indexing, and maintenance for the heterogeneous processors and shared memory of Snapdragon 8-series platforms. MobileRAG [111] implements a fully on-device retrieval-augmented generation pipeline through partitioned index loading and post-retrieval reduction. Both demonstrate smartphone execution of the reported indexing or retrieval pipeline. MUSE evaluates multimedia retrieval, whereas MobileRAG focuses on document-based retrieval-augmented generation. Neither result alone establishes a complete personalized agent-memory pipeline.

6.1.2 Laptops and Desktops. Laptops and desktops can support stronger local models, but their windows, webpages, and documents often belong to several interleaved tasks. As activity histories

grow, the first formation decision is which observations should be retained and which task they belong to. FOCAL [190] filters desktop observations before visual reasoning and uses a local 8B model on an Apple M4 laptop to construct task-specific logs and summaries. Desktop memory therefore has to identify task boundaries as well as control how much information it stores.

When memory expands to long-lived document collections, index storage becomes a separate concern. LEANN [163] recomputes selected embeddings during retrieval to reduce the amount of index data that must remain in storage. This design trades additional computation for a smaller persistent footprint, allowing larger indexes to be maintained on personal systems. Desktop histories may also contain application experience that can be reused across tasks. ActionEngine [208] represents the states and transitions of a web application in an updatable model and revises that model after execution failures. Such updating allows procedural experience to evolve with the application instead of being relearned whenever the interface changes. Desktop memory is also being explored for game interaction, including compact plans in PUBG Ally [115] and separate NPC dialogue stores in fixed-persona language models [19].

6.2 Head-Worn Systems

Head-worn systems observe the environment from a first-person perspective. For smart glasses, many events cannot be recaptured after they pass. Memory therefore begins with selective sensing and retention. The retained observations are then organized into personal experience for later interaction. XR headsets present a related but shorter-horizon problem, in which spatial and interaction context should be maintained within an active session.

6.2.1 Smart Glasses. Smart glasses are expected to support prolonged first-person sensing within a face-worn form factor. This combination places cameras, onboard computation, and wireless transmission within tight battery, thermal, and bandwidth budgets. Some research in this setting seeks to reduce high-cost visual data before it moves through the full processing pipeline. EgoTrigger [112] activates the camera from low-cost audio cues. ColorTrigger [21] preserves a grayscale stream and acquires color only when it is expected to help.

The resulting records follow the wearer’s viewpoint through everyday life. Relevant people, objects, and events may disappear from view and reappear hours or days later. EgoLife [183] captures week-long multimodal activity with Meta Aria glasses and evaluates questions that span extended periods. EgoGraph [142] links recurring entities and events across time in long egocentric video. EgoLife provides cross-day records and evaluation tasks, while EgoGraph contributes a temporal graph for organizing recurring entities and events.

The hands-free, situated form factor also changes how memory output is encountered. Assistance can arrive while the user is physically occupied, without a deliberate switch to another application or device. ProMemAssist [114] evaluates proactive support during tasks that combine physical action with cognitive demand. A study of personalized proactive agents [45] examines reactions to proactive behaviors presented through first-person smart glasses scenarios. In this setting, recalled information enters the same perceptual and attentional space as the ongoing activity. The resulting design problem is to make relevant past information available without drawing the user away from the physical activity in which it is needed.

How these functions are distributed across hardware remains a separate deployment question. Current smart glasses prototypes place sensing and lightweight selection closest to the glasses, whereas memory construction and higher-level reasoning are generally handled externally. EgoTrigger [112] validates selective capture on embedded hardware. EgoLife [183] supplies data from a

wearable setting, and EgoGraph [142] organizes entities and events in long video. Their memory construction and querying are evaluated offline. ProMemAssist [114] combines smart glasses sensing with processing on a companion laptop.

6.2.2 XR Headsets. Unlike the cross-day experience targeted by smart glasses memory, XR research usually maintains spatial, object, and interaction context within an active session. Such context helps a system resolve underspecified references and combine gaze or pointing with speech to infer the user's intent. Memory here functions primarily as session-level environmental state. ClickAIXR [68] retains interaction context around an object selected by the user to support subsequent speech and visual processing. SpeechLess [71] combines earlier spatial, temporal, activity, and referential cues to interpret incomplete spoken instructions. PITAR [60] integrates speech, gaze, and pointing to maintain scene and interaction state. All three use the session's environmental state to reduce how much spatial and object information the user must repeat.

6.3 Personal Sensing Systems

Personal sensing systems build usable context from intermittent physiological, motion, or acoustic signals. Their records differ from those collected by systems centered on dense visual or interface data. Smartwatch systems use these signals to identify events or track ongoing procedures. Health and assistive systems retain longer-term records to guide interpretation, intervention, and recall of everyday activities.

6.3.1 Smartwatches. Continuous smartwatch operation is first constrained by sampling and sensor power. SAMoSA [105] uses motion signals to locate candidate activity periods and then analyzes heavily subsampled audio to retain additional activity cues at a lower sensing cost. Although SAMoSA does not implement a complete agent memory system, its low-cost trigger determines which everyday events can enter subsequent memory processing. For procedural assistance, memory instead tracks progress through an ongoing activity. PrISM-Observer [7] follows the steps of everyday procedures and predicts when an omitted or upcoming step warrants intervention. Its short-term memory is a procedure state that evolves with the user's actions. The full local execution in Table 6-1 refers to the watch-only implementation, while the user study used a separate laptop-server implementation.

6.3.2 Health and Assistive Devices. Health and assistive systems use memory for both longitudinal interpretation and support during everyday activity. PHIA [103] combines longitudinal wearable records with retrieval and code execution to produce personalized health insights. ECG-Agent [33] evaluates the feasibility of 1B and 3B models for multi-turn electrocardiogram analysis and tool use. These experiments used RTX A6000 GPUs, leaving execution of the complete system on the target mobile hardware unverified.

Affective Agent [106] connects intervention decisions to locally stored episodic and semantic memory. Recent intervention outcomes and accumulated preferences inform later decisions, while the host software validates proposed memory updates before storing them. Its compact, memory-conditioned model is evaluated on Raspberry Pi CM5 as an embedded platform for wearable systems.

Memory can also provide direct assistance with recalling everyday actions. MemPal [100] uses a neck-worn smartphone to maintain a local activity diary, with cloud models interpreting observations and answering spoken object-location queries. A study in the homes of 15 older adults found that audio assistance reduced the number of rooms searched. The system connects retained personal events to a concrete task performed in the environment where those events occurred.

6.4 Embodied Systems

In embodied systems, stored maps, object states, and skills become part of the perception–action loop by informing navigation and manipulation. For physical robots, the key deployment question is which stages of memory maintenance, retrieval, and memory-conditioned planning run on the robot itself. Simulation provides a complementary setting for examining retention, compression, and skill reuse over long task horizons.

6.4.1 Physical Robots. Studies on robotic platforms combine execution on the robot and validation in the physical world in different ways. VLONS [129] runs its mapping, target understanding, and planning pipeline locally on Jetson hardware and evaluates it across several physical robot platforms. EfficientNav [185] runs memory retrieval and planning on Jetson, with navigation evaluated mainly in Habitat. STaR [192] uses a physical Husky robot to collect environmental information and constructs its long-term multimodal memory on an external RTX 3080 laptop. On-robot execution, physical-world evaluation, and coverage of the memory pipeline should therefore be reported as separate dimensions.

6.4.2 Simulated Environments. Simulation provides controlled task horizons and environmental variation, making it useful for isolating what an embodied agent should retain. MemCtrl [42] learns to retain, update, or discard observations and reflections during exploration. Memo [52] periodically compresses prior experience into separately stored summary states, reducing the amount of history carried by a reinforcement learning policy. Selective retention and periodic compression offer two ways to control memory growth over long tasks. In simulation, embodied agents may also accumulate reusable skills or knowledge about personally meaningful objects alongside changing environmental state. Voyager [150] accumulates successful experience as executable skills in Minecraft. MEMENTO [75] examines how personal meanings attached to objects and recurring behavior patterns affect planning with a simulated Spot robot.

7 EVALUATING ON-DEVICE AGENT MEMORY: DIMENSIONS AND BENCHMARKS

Rigorous evaluation is essential for determining whether on-device agent memory can support sustained interaction. Existing studies adopt diverse tasks and metrics because both memory demands and operating constraints vary across device contexts. We organize this evidence around four dimensions: memory accuracy, lifecycle management, personalization quality, and on-device runtime cost. We then examine the general agent-memory benchmarks and device-context benchmarks used to measure them.

7.1 Evaluation Dimensions

7.1.1 Memory Accuracy. Early evaluations of long-term memory centered on whether an agent could recover information from its interaction history. LoCoMo [97] broadens this setting by pairing long-term question answering with event summarization and multimodal dialogue generation. LongMemEval [170] organizes questions around evidence extraction, cross-session and temporal reasoning, history updates, and abstention. By tracing errors across indexing, retrieval, and reading, it helps separate access failures from reasoning failures. Such benchmarks typically combine answer accuracy, model-based judgments, and evidence recall.

Some benchmarks isolate memory quality from final task outcomes. MemBench [143] evaluates factual and reflective memory under participatory and observational interaction settings. MemoryAgentBench [59] presents history incrementally, allowing retrieval, test-time learning, and long-context understanding to be assessed within one interaction protocol. In mobile environments,

Memory-World [154] reports the accuracy of explicit memory separately from task success, so memory errors can be distinguished from downstream action errors.

When remembered information constrains tool use or environment interaction, accuracy must also be measured at the action level. Mem2ActBench [127] distributes task requirements across long dialogues and scores both tool selection and argument completion. MemoryArena [53] links tasks across sessions so earlier actions and feedback alter later requirements. AMA-Bench [204] uses long-horizon agent trajectories to test state tracking, causal reasoning, and experience abstraction. These settings connect memory accuracy to task success, action validity, and repeated-error patterns.

7.1.2 Lifecycle Management Effectiveness. Lifecycle evaluation asks whether memory remains reliable through memory formation, retrieval, consolidation, and selective forgetting. Because evidence arrives over time, evaluation protocols must preserve the order in which information becomes available. S-EMBER [161] releases egocentric events as a stream and evaluates both the answer and its supporting time span. LifeDialBench [206] restricts each query to memory formed beforehand, creating an online protocol for continuous lifelogging. WorldMemArena [85] scores formation, updating, retrieval, and use separately, exposing failures at different lifecycle stages.

The timing of a query also matters when evaluating memory compression. PA-Bench [213] withholds the recall question until cache selection is complete and measures retained fact blocks alongside answer accuracy. On synthetic personal conversations, its comparisons show how query-informed selection can overstate the ability to retain facts for unknown future questions. This protocol tests whether useful information survives compression before the system knows which part will be needed.

Once new evidence arrives, evaluation shifts to consolidation and updating. MemoryBench [3] reveals task feedback incrementally and tests whether execution outcomes become reusable experience. PERMA [90] and Memora [147] introduce changing preferences and obsolete facts, distinguishing the adoption of new evidence from continued use of stale memory. PM-Bench [86] extends temporal change to prospective memory by testing whether plans are retained, revised, canceled, or triggered when conditions are met. Useful measures include post-update consistency, stale-memory use, transfer across tasks, and missed or false triggers.

Forgetting tests whether target information stops influencing the system while unrelated memory remains available. MemoryAgentBench [59] includes selective forgetting in incremental interactions and measures both outcomes. Tests of deployment-time memorization [30] follow deleted source records into derived summaries, extending verification beyond the source record.

7.1.3 Personalization Quality. Personalization evaluation asks whether personal history improves responses and actions for the current user. MOBIMEM [92] derives user facts, preferences, and experience from prior tasks. It then tests whether an agent can recover omitted conditions and use them during execution. AndroidIntent, introduced with PersonalAlign [96], builds ambiguous requests and proactive assistance tasks from long Android histories. It evaluates implicit preference recovery and recognition of user routines. Relevant outcomes include profile alignment, task success, recovered user constraints, and the quality of proactive assistance.

Personal memory must also be used selectively. OP-Bench [58] treats irrelevant recall, repeated personalization, and sycophantic responses as distinct forms of overpersonalization. MemPrivacy-Bench [31] organizes cases by sensitivity and jointly measures privacy preservation, task utility, and processing latency. Appropriate-use rates, leakage rates, and personalized task performance therefore capture complementary aspects of user alignment.

7.1.4 On-Device Runtime Cost. Resource evaluation should distinguish direct measurements from indicators of computational workload. MobileRAG [111] reports retrieval-stage memory use, time

to first token, and energy per query on a Galaxy S24. MUSE [202] measures retrieval throughput and index construction time on mobile SoCs, with energy measurements covering the construction interval. FOCAL [190] characterizes desktop processing through model-call counts and token consumption, while leaving latency and energy measurements to future work. BudgetBench [118] complements these studies by checking context-budget feasibility and accounting failures at individual agent calls.

Measurements of individual stages reveal where resources are spent, but their implications depend on how those stages interact. A shorter response time may, for example, rely on additional storage or work performed between requests. Assessing the budgets in Section 2.2.3 therefore requires examining query processing and memory maintenance together under the same operating conditions. Tracking how this balance changes as histories grow helps determine whether improvements in individual components support sustained on-device operation.

7.2 Benchmarks

Available evaluation resources fall into general agent-memory benchmarks and device-context benchmarks. General benchmarks offer controlled dialogue histories, event streams, or agent trajectories, enabling methods to be compared under common inputs. Device-context benchmarks introduce personal records, continuous sensing, or device interaction tasks to expose memory demands that arise in use. For on-device methods, the two groups are complementary. General benchmarks provide a cross-system capability reference, while device-context benchmarks capture performance under particular hardware and interaction conditions.

Table 7-1 summarizes the scenarios and main evaluation focus of representative resources.

Table 7-1. General and device-context benchmarks for agent memory

Benchmark	Scenario	Evaluation focus	Link
General			
LoCoMo [97]	Dialogue	Recall across dialogue sessions	Link
LongMemEval [170]	Dialogue	Recall and reasoning over evolving dialogue histories	Link
MemBench [143]	Agent interaction	Factual and reflective memory	Link
MemoryAgentBench [59]	Agent interaction	Retrieval and learning from incremental history	Link
MemoryBench [3]	Agent tasks	Learning from task feedback	Link
WorldMemArena [85]	Agent interaction	Memory maintenance and use during interaction	Link
Mem2ActBench [127]	Tool use	Tool use with remembered constraints	Link
MemoryArena [53]	Agent tasks	Task completion using experience from earlier sessions	Link
Memora [147]	Dialogue	Updating outdated personal information	Link
PM-Bench [86]	Personal assistance	Remembering intentions and acting when triggered	Link
LongMemEval-V2 [169]	Web interaction	Recall of environmental knowledge and experience	Link
AMA-Bench [204]	Agent tasks	State tracking and causal reasoning over agent histories	Link
PA-Bench [213]	Personal assistance	Fact retention after compression for unknown queries	—
BudgetBench [118]	Agent tasks	Compliance with context budgets at each agent call	Link
Device context			
MobileMem [39]	Mobile applications	Recall across application histories	Link
MobileMem-Omni [40]	Mobile applications	Recall across multimodal histories	Link
MEMARENA [196]	Dialogue	Recall with access control	—
Memory-QA [61]	Egocentric records	Retrieval of events by time and place	Link
S-EMBER [161]	Egocentric records	Event recall and temporal localization in video streams	Link

Table 7-1 (continued)

Benchmark	Scenario	Evaluation focus	Link
LifeDialBench [206]	Lifelogging	Recall from continuous dialogue	Link
MemGUI-Bench [87]	Mobile GUI	Information retention and reuse across GUI interactions	Link
AndroidWorld-M [154]	Mobile GUI	Task completion using past information	—
Memory-World [154]	Mobile GUI	Retention of transient information	—
DataScope [84]	Mobile GUI	Tracking information across applications	—
AndroidIntent [96]	Mobile GUI	Inference of implicit user intent from personal history	Link
MobileIAR [173]	Mobile GUI	Inference of implicit user intent from demonstrations	Link
PSPA-Bench [107]	Mobile GUI	Planning with user preferences	—
DesktopBench [190]	Desktop activity	Summarization of desktop activity	Link
HME-QA [112]	Egocentric records	Recall of past events with limited capture	Link
EgoLifeQA [183]	Egocentric records	Recall and integration of events across multiple days	Link
M3-Bench [93]	Video understanding	Reasoning over long videos with multimodal evidence	Link
MENTO [75]	Embodied tasks	Task completion using personal knowledge	Link

7.2.1 General Agent-Memory Benchmarks. Multi-session dialogue provides the most established structure for general memory benchmarks. LoCoMo [97] uses extended conversations to test factual recall, cross-record integration, and temporal relations. Event summarization and multimodal generation broaden the task beyond question answering. LongMemEval [170] constructs evidence-controlled questions around locating information, reasoning across sessions, and handling changes in history. It includes abstention cases and histories of varying lengths. Human-authored questions and evidence are embedded in model-expanded, manually revised conversations with substantial distractor content. This design requires systems to recover dispersed evidence over long time spans.

Other benchmarks treat memory as an online process rather than a fixed context. MemBench [143] combines user profiles, multi-session records, and question answering under participatory and observational settings. MemoryAgentBench [59] converts existing long-context datasets and newly constructed tasks into round-by-round interactions, so agents receive history incrementally. MemoryBench [3], WorldMemArena [85], Memora [147], and PM-Bench [86] organize instances as time-ordered sequences. New evidence can revise earlier memory, and each test query or action depends only on history available at that point.

Task-oriented benchmarks use tool calls, linked task episodes, or full agent trajectories as the memory source. Mem2ActBench [127] constructs tool-use cases whose constraints are distributed across earlier dialogue. MemoryArena [53] preserves dependencies between tasks completed in different sessions. LongMemEval-V2 [169] and AMA-Bench [204] retain environment state and interaction history in agent trajectories, then query environmental knowledge or past experience.

Taken together, general benchmarks place memory in long-horizon settings where histories evolve and later tasks depend on earlier interaction. Success requires agents to preserve relevant evidence, incorporate feedback, and carry experience into subsequent answers or actions. Their shared task structures support method comparison and provide a broad capability reference for on-device systems.

7.2.2 Device-Context Benchmarks. Device-context benchmarks draw on the data streams and interaction structures associated with particular device form factors. For mobile GUI agents, AndroidWorld [120] is a widely reused programmable environment. It generates tasks across Android applications from parameterized instructions and verifies success against application state. Several mobile agent memory studies adopt this environment [84, 92, 141, 176, 207]. STAMP evaluates its trained agent on memory-dependent tasks in AndroidWorld-M [154]. Memory-specific benchmarks

encode historical dependence directly. MemGUI-Bench [87] requires agents to retain information across page, application, or trial boundaries. Memory-World [154] and DataScope [84] use transient variables or distractor-rich cross-application objects as the information to retain. By marking where past information becomes relevant, these resources connect earlier screens to later actions.

Another family of mobile resources retains longer personal application histories. MobileMem [39] uses volunteer profiles and phone-use metadata to generate de-identified application content, organizing records and user-assistant dialogue into continuous trajectories. The later MobileMem report [40] extends this setting with MobileMem-Omni. Persona and phone-use priors guide the synthesis of year-scale histories, while bilingual dialogue and generated images introduce textual and visual evidence across applications. These complementary settings support evaluation of recall, temporal reasoning, knowledge updating, and implicit preferences. AndroidIntent [96], MobileIAR [173], and PSPA-Bench [107] pair personal histories or demonstrations with later mobile tasks, thereby embedding implicit user information in action-oriented instances.

Desktop benchmarks organize data around continuous screenshots and task sessions. DesktopBench [190] reconstructs activity sessions from VideoGUI, including interleaved tasks and interrupted tasks that later resume. Each screenshot is labeled for task membership, retention value, and summary content. A session therefore preserves both interface changes and task switching.

Head-worn benchmarks are built primarily from egocentric video and image streams. Ego4D [50] provides a large-scale collection of first-person video and an episodic-memory task that localizes answer-bearing time segments. Later benchmarks use egocentric records for long-video question answering and for evaluating sparse capture. HME-QA [112] and Memory-QA [61] add capture-cost constraints or spatiotemporal image metadata. EgoLife extends this setting across days through multimodal, multi-view recordings from Meta Aria glasses. EgoLifeQA [183] asks about daily events and longer-term patterns, sometimes requiring evidence from different days. EgoGraph [142] and EgoMemory [145] reuse these records for entity-relation modeling and experience-retrieval tasks.

Personal sensing systems generate continuous health time series rather than interface histories. PHIA [103] builds synthetic user records from large Fitbit and Pixel Watch datasets. Its questions combine automatically scored analysis with open-ended interpretation of trends and personal behavior.

Embodied benchmarks use visual observations and long task trajectories. M3-Bench [93] pairs long video records with questions that require multiple pieces of evidence and cross-modal reasoning. MEMENTO [75] links memory-acquisition episodes to personalized tasks in simulated homes. STaR [192] constructs questions about spatial, object, and event information from navigation histories. All three require agents to use earlier observations in later tasks.

Overall, device-context benchmarks embed memory evaluation in the data streams and interaction patterns associated with particular form factors. Whether the source is a GUI trace or a continuous sensor stream, the agent must retain goal-relevant evidence and reuse it in later responses or actions. These resources reveal how memory requirements change across devices and can guide design choices for memory formation, retrieval, and updating.

8 OPEN CHALLENGES AND FUTURE DIRECTIONS

Advances in on-device memory are expanding how agents draw on personal experience, yet important limitations remain in sustained everyday use. Understanding these limitations requires attention to interactions across the memory lifecycle and to the conditions under which existing capabilities remain effective.

8.1 Resource-Aware Lifecycle Management

On-device memory shares limited compute and memory capacity with model inference, while ongoing maintenance adds work beyond individual queries [111, 202]. Moving costly consolidation out of the response path already reduces the burden of immediate interaction [46]. Extending this approach to sustained operation on a device requires maintenance schedules to follow changes in resource availability while keeping updates timely. Whether such coordination delivers lasting gains depends on how it behaves as both the memory and the workload evolve.

Task feedback has also made it possible to learn joint policies for long-term memory and active context, linking retention decisions to downstream outcomes [191]. Sparse, delayed rewards complicate credit assignment when several memory decisions shape the same outcome. Future work should establish how much history such learning needs and whether its benefits justify the cost of training and running the manager.

Further directions include wear-aware memory placement [29] and recurrent-state restoration of persistent context [49]. Their value needs to be established through sustained agent workloads on target devices.

8.2 Reliable Memory Evolution

Reliability depends on how well memory preserves the evidence behind its claims as that memory evolves. Existing systems retain the temporal history of facts or separate source records from editable memory, allowing later judgments to be checked against their evidence [11, 184]. Access to a source does not, however, establish that successive summaries remain faithful to its meaning. Evaluations of continuous lifelogs show that lossy compression can remove context needed for later answers [206].

A further concern arises when one summary becomes evidence for the next. If an unverified interpretation survives successive revisions and appears in several records, their agreement may be mistaken for independent corroboration. Reliability checks need to follow these dependencies across revisions and distinguish independent support from repeated use of the same source.

Editable personal memory already gives users a way to correct individual records [166]. Keeping the rest of the memory consistent with those corrections requires a broader view of which summaries and inferences depend on each claim. On devices that cannot retain every source indefinitely, an open question is which evidence and links are sufficient to support later verification and repair.

8.3 Continual Personalization

A user's choices acquire meaning from the circumstances in which they are made. Clarification before action and correction afterward already help agents fill information gaps and revise preferences [83]. Temporal benchmarks nevertheless show that preference inference remains difficult across long event sequences and interaction domains [90]. In everyday interaction, feedback may leave unclear whether a user is rejecting an action or revising a lasting preference. Continual personalization needs to carry this uncertainty forward, allowing later evidence to revise the judgment. If an agent filters its actions through existing preferences, it may also restrict the evidence available to revise its understanding of the user.

Even an accurate account of a preference does not establish its relevance to the current task. Benchmarks of overpersonalization show that unnecessary use of personal history can pull a response away from the user's request [58]. Work on proactive assistance likewise considers the value of help alongside the cost of interruption [114]. Future systems need to separate a preference's relevance to the present task from evidence that the preference itself has changed. Clarification

can resolve some uncertainty, but its value depends on the decisions it improves and the attention it asks of the user.

8.4 Privacy Protection in Memory Use

Local storage and safeguards on outgoing information already provide ways to limit the exposure of personal memory [13, 31]. Research on interactions with third parties nevertheless shows that manipulated task contexts can elicit disclosures unrelated to the user’s request [10]. For persistent memory, restrictions attached only to a source record may leave its summaries and inferences without clear authorization for later use. Combining records may also reveal sensitive information that restrictions on individual records do not adequately cover. An open question is how to carry these restrictions through derivation while reassessing permitted use as tasks and purposes change.

8.5 Verifiable Forgetting

Experiments that extend deletion from source records to derived summaries have reduced measured textual residue to zero within the tested settings [30]. They provide a basis for coordinated deletion, but evidence for other representations and residual effects must be established separately.

When an experience is retained in explicit memory and also used for model adaptation, erasure must address both the stored record and its learned influence. The scope of a request needs to be made explicit, with checks for residual effects that also assess preservation of unrelated capabilities. Existing execution proofs can verify that an agreed model update was performed, but they do not establish that the target information has ceased to influence the model [98]. Likewise, a failed recovery attempt supports conclusions only within the conditions and probes used. Verification should relate the operations performed to the effects actually assessed, so that a claim of forgetting has a clear and testable scope.

Users also need an intelligible account of what a deletion request achieved. Yet the records kept to explain that outcome may themselves retain sensitive information. An audit trail should make the outcome open to inspection without creating another repository for the information the user asked to remove.

8.6 Long-Term Evaluation on Real Devices

Recent benchmarks have extended evaluation to connected tasks and changing preferences, with some explicitly preserving the order of information arrival [53, 90, 206]. Controlled temporal tasks can compare memory capabilities under the same history. In continuing interaction, memory also shapes the actions an agent takes and, through those actions, the information it encounters next. Diverging trajectories are therefore part of the effect to be understood. Connecting comparisons under fixed histories with studies of sustained interaction could show which memory decisions deliver lasting benefits, and how those benefits change with use.

Device experiments already provide measurements of retrieval, generation, and index maintenance under specific workloads [111, 202]. Sustained studies can reveal how these costs accumulate as memory grows and how background work affects ordinary interaction. They should also track the time users spend correcting and managing memory, asking whether this burden declines as the system adapts.

9 CONCLUSION

This survey has brought together research on on-device agent memory, spanning its foundations, representations, lifecycle management, personalization, deployment, and evaluation. Comparing textual, vector, KV-cache, graph-structured, and parametric memory clarifies how their basic units shape trade-offs in access, updating, and resource use. Extending the analysis to formation,

retrieval, evolution, and forgetting connects memory use to the work of keeping information available and current within device budgets. In personalized applications, memory connects past experience with present needs through preference learning, personal reference resolution, and proactive assistance. Privacy safeguards and user control govern the retention and use of personal information throughout these applications. Deployment and evaluation studies connect these designs to specific devices and tasks, revealing where memory runs and which resource demands have been measured.

As personal histories grow and user needs change, further evidence is needed on memory reliability through updating and forgetting. Long-term studies on real devices could help assess whether its benefits for later tasks can be sustained within available resource budgets. We hope the taxonomy and evidence assembled here help researchers and practitioners compare designs, assess their applicability, and develop reliable on-device memory under user control.

REFERENCES

- [1] Junyong Ahn, Hyeongrok Han, Bong Gyun Kang, Jisoo Mok, Byunghan Lee, and Sungroh Yoon. 2026. MobileKGQA: On-Device KGQA System on Dynamic Mobile Environments. In *International Conference on Learning Representations*. https://proceedings.iclr.cc/paper_files/paper/2026/hash/24e4e3234178a836b70e0aa48827e0ff-Abstract-Conference.html
- [2] Kwangseob Ahn. 2025. HEMA : A Hippocampus-Inspired Extended Memory Architecture for Long-Context AI Conversations. arXiv:2504.16754 [cs.CL] <https://arxiv.org/abs/2504.16754>
- [3] Q. Ai, Y. Tang, C. Wang, J. Long, W. Su, and Y. Liu. 2026. MemoryBench: A Benchmark for Memory and Continual Learning in LLM Systems. *Proceedings of the 43rd International Conference on Machine Learning*. <https://arxiv.org/abs/2510.17281>
- [4] Keivan Alizadeh, Seyed Iman Mirzadeh, Dmitry Belenko, S. Khatamifard, Minsik Cho, Carlo C. Del Mundo, Mohammad Rastegari, and Mehrdad Farajtabar. 2024. LLM in a flash: Efficient Large Language Model Inference with Limited Memory. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 12562–12584. <https://doi.org/10.18653/v1/2024.acl-long.678>
- [5] S. Alqithami. 2025. Forgetful but Faithful: A Cognitive Memory Architecture and Benchmark for Privacy-Aware Generative Agents. arXiv preprint arXiv:2512.12856. <https://arxiv.org/abs/2512.12856>
- [6] E. Aman, E. Carlin, H.-K. Pao, G. Beltrame, G. I. P. Sari, and Y.-T. Chen. 2025. BitMar: Low-Bit Multimodal Fusion with Episodic Memory for Edge Devices. *Proceedings of the BabyLM Workshop at EMNLP 2025*. <https://arxiv.org/abs/2510.10560>
- [7] R. Arakawa, H. Yakura, and M. Goel. 2024. PrISM-Observer: Intervention Agent to Help Users Perform Everyday Procedures Sensed using a Smartwatch. *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, 1-16. <https://doi.org/10.1145/3654777.3676350>
- [8] Mustafa Arslan. 2026. Nexus: Depth-Adaptive KV-Cache Splicing and Retrieval-Decoupled Tool Routing for Agentic LLMs on Unified Memory. arXiv preprint arXiv:2608.20397. arXiv:2608.20397 <https://arxiv.org/abs/2608.20397>
- [9] Alan Baddeley. 2003. Working memory: looking back and looking forward. *Nature Reviews Neuroscience* 4, 10 (2003), 829–839. <https://doi.org/10.1038/nrn1201>
- [10] Eugene Bagdasarian, Ren Yi, Sahra Ghalebikesabi, Peter Kairouz, Marco Gruteser, Sewoong Oh, Borja Balle, and Daniel Ramage. 2024. AirGapAgent: Protecting Privacy-Conscious Conversational Agents. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. ACM, 3868–3882. <https://doi.org/10.1145/3658644.3690350>
- [11] P. Banerjee, M. Moshtaghi, S. Subramanian, A. Misra, and A. Chadha. 2026. APEX-MEM: Agentic Semi-Structured Memory with Temporal Reasoning for Long-Term Conversational AI. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 16470–16489). <https://doi.org/10.18653/v1/2026.acl-long.749>
- [12] J. Bang, J. Lee, K. Shim, S. Yang, and S. Chang. 2024. Crayon: Customized On-Device LLM via Instant Adapter Blending and Edge-Server Hybrid Inference. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 3720–3731). <https://doi.org/10.18653/v1/2024.acl-long.204>
- [13] V. P. Bhardwaj. 2026. SuperLocalMemory: Privacy-Preserving Multi-Agent Memory with Bayesian Trust Defense Against Memory Poisoning. arXiv preprint arXiv:2603.02240. <https://arxiv.org/abs/2603.02240>
- [14] Varun Pratap Bhardwaj. 2026. SuperLocalMemory V3.3: The Living Brain – Biologically-Inspired Forgetting, Cognitive Quantization, and Multi-Channel Retrieval for Zero-LLM Agent Memory Systems. arXiv preprint arXiv:2604.04514.

arXiv:2604.04514 <https://arxiv.org/abs/2604.04514>

- [15] Varun Pratap Bhardwaj, Garima Singh, and Arun Pratap Bhardwaj. 2026. SuperLocalMemory 4.0: The Governed Memory Operating System for AI Agents. arXiv preprint arXiv:2608.08253. arXiv:2608.08253 <https://arxiv.org/abs/2608.08253>
- [16] M. Bini, O. Bohdal, U. Michieli, Z. Akata, M. Ozay, and T. Ceritli. 2025. MemLoRA: Distilling Expert Adapters for On-Device Memory Systems. arXiv preprint arXiv:2512.04763. <https://arxiv.org/abs/2512.04763>
- [17] O. Bohdal, P. Saha, U. Michieli, M. Ozay, and T. Ceritli. 2026. Clustering-driven Memory Compression for On-device Large Language Models. arXiv preprint arXiv:2601.17443. <https://arxiv.org/abs/2601.17443>
- [18] Sayed Pedram Haeri Boroujeni, Niloufar Mehrabi, Patrick Woods, Gabriel Hillesheim, and Abolfazl Razi. 2026. Don't Waste Bits! Adaptive KV-Cache Quantization for Lightweight On-Device LLMs. arXiv:2604.04722 [cs.CV] <https://arxiv.org/abs/2604.04722>
- [19] Martin Braas and Lukas Esterle. 2025. Fixed-Persona SLMs with Modular Memory: Scalable NPC Dialogue on Consumer Hardware. arXiv preprint arXiv:2511.10277. arXiv:2511.10277 <https://arxiv.org/abs/2511.10277>
- [20] Hongru Cai, Yongqi Li, Ran Wei, and Wenjie Li. 2026. PalmClaw: A Native On-Device Agent Framework for Mobile Phones. arXiv preprint arXiv:2607.13027. arXiv:2607.13027 <https://arxiv.org/abs/2607.13027>
- [21] W. Cai, H. Zhang, Y. Huang, S. Sun, J. Deng, S. Xu, J. Song, and Z. Zhang. 2026. Color When It Counts: Grayscale-Guided Online Triggering for Always-On Streaming Video Sensing. arXiv preprint arXiv:2603.22466. <https://arxiv.org/abs/2603.22466>
- [22] M. C. Cakmak, N. Agarwal, and D. Poudel. 2025. TriPSS: A Tri-Modal Keyframe Extraction Framework Using Perceptual, Structural, and Semantic Representations. Proceedings of the MA-LLM Workshop at ACM Multimedia 2025. <https://arxiv.org/abs/2506.05395>
- [23] Shuqi Cao, Jingyi He, and Fei Tan. 2026. HiGMem: A Hierarchical and LLM-Guided Memory System for Long-Term Conversational Agents. arXiv:2604.18349 [cs.CL] <https://arxiv.org/abs/2604.18349>
- [24] T. Ceritli, O. Bohdal, M. Ozay, J. Moon, K. Lee, H. Ko, and U. Michieli. 2025. HydraOpt: Navigating the Efficiency-Performance Trade-off of Adapter Merging. In Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (pp. 26887–26909). <https://doi.org/10.18653/v1/2025.emnlp-main.1365>
- [25] Jiale Chang and Yuxiang Ren. 2026. ScrapMem: A Bio-inspired Framework for On-device Personalized Agent Memory via Optical Forgetting. arXiv preprint arXiv:2605.03804. arXiv:2605.03804 <https://arxiv.org/abs/2605.03804>
- [26] Gong Chen, Mingkai Lin, Xiaobin Hong, Wenzhong Li, and Sanglu Lu. 2026. EcoTune: Edge-Cloud Collaborative Model Adaptation for Budget-Constrained On-Device SLM Personalization. In *Proceedings of the ACM Web Conference 2026*. Association for Computing Machinery, 11 pages. <https://doi.org/10.1145/3774904.3792264>
- [27] J. Chen, Y. Li, and G. Wang. 2026. MemFlow: Intent-Driven Memory Orchestration for Small Language Model Agents. arXiv preprint arXiv:2605.03312. <https://arxiv.org/abs/2605.03312>
- [28] Jiaqi Chen, Yongqin Zeng, Shaoshen Chen, Yijian Zhang, Hai-Tao Zheng, Chunxia Ma, and XiuTeng Zhou. 2026. CoreMem: Riemannian Retrieval and Fisher-Guided Distillation for Long-Term Memory in Dialogue Agents. arXiv:2606.18406 [cs.CL] <https://arxiv.org/abs/2606.18406>
- [29] Josef Liyanjun Chen. 2026. Memory as a Wasting Asset: Pricing Flash Endurance for Embodied Agents, and the Limits of Doing So. arXiv preprint arXiv:2606.18144. arXiv:2606.18144 <https://arxiv.org/abs/2606.18144>
- [30] L. Chen, G. Zhang, K. Zhao, D. Cirne, A. Olsen, X. Chu, Z. Miller, A. Blanken, A. Anoun, and J. Ting. 2026. Deployment-Time Memorization in Foundation-Model Agents. ICML 2026 Workshop on Memorization in Foundation Models. <https://arxiv.org/abs/2606.10062>
- [31] Y. Chen, J. Zhao, B. Tang, H. Wang, Y. Zhang, F. Huang, F. Xiong, and Z. Li. 2026. MemPrivacy: Privacy-Preserving Personalized Memory Management for Edge-Cloud Agents. arXiv preprint arXiv:2605.09530. <https://arxiv.org/abs/2605.09530>
- [32] P. Chhikara, D. Khant, S. Aryan, T. Singh, and D. Yadav. 2025. Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory. arXiv preprint arXiv:2504.19413. <https://arxiv.org/abs/2504.19413>
- [33] H. Chung, J. Oh, D. Kyung, J. Kim, Y. Kwon, M.-G. Kim, and E. Choi. 2026. ECG-Agent: On-Device Tool-Calling Agent for ECG Multi-Turn Dialogue. Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing. <https://arxiv.org/abs/2601.20323>
- [34] J. Chung, B. Shin, J. Kim, and M. Rhu. 2026. Agent-X: Full Pipeline Acceleration of On-device AI Agents. arXiv preprint arXiv:2605.10380. <https://arxiv.org/abs/2605.10380>
- [35] Gabriele Conte, Alessio Mattiace, Gianni Carmosino, Potito Aghilar, Giovanni Servedio, Francesco Musicco, Vito Walter Anelli, Tommaso Di Noia, and Francesco Maria Donini. 2026. RUVA: Personalized Transparent On-Device Graph Reasoning. arXiv preprint arXiv:2602.15553. arXiv:2602.15553 <https://arxiv.org/abs/2602.15553>
- [36] Nelson Cowan. 2008. What are the differences between long-term, short-term, and working memory? In *Essence of Memory*. Progress in Brain Research, Vol. 169. Elsevier, 323–338. [https://doi.org/10.1016/S0079-6123\(07\)00020-9](https://doi.org/10.1016/S0079-6123(07)00020-9)

- [37] Lipeng Dai, Luping Xiang, and Kun Yang. 2026. A token/KV-cache communication media selection and resource allocation strategy for multi-agent collaboration. *Science China Information Sciences* (2026). <https://doi.org/10.1007/s11432-026-5057-2>
- [38] Darwin Agent Team. 2026. Mi-Memory: A Lifecycle Memory Framework for Personal AI. arXiv preprint arXiv:2607.18975. arXiv:2607.18975 <https://arxiv.org/abs/2607.18975>
- [39] X. Deng, Y. Xue, Y. Chen, M. Mao, R. Zhong, B. Xu, J. Fang, H. Xu, T. Wu, Y. Xu, S. Deng, H. Wang, H. Chen, and N. Zhang. 2026. MobileMem: Evaluating Long-Horizon Memory for Language Agents in Real-World Mobile Environments. Lifelong Agent Workshop at ICLR 2026. <https://openreview.net/forum?id=w5I1HrMgJ>
- [40] Xinle Deng, Yida Xue, Xiangyuan Ru, Yijun Chen, Buqiang Xu, Mingjun Mao, Xinjie Liu, Haoming Xu, Shuofei Qiao, Mengru Wang, Chen Jiang, Yuchen Eleanor Jiang, Lizhong Wang, Jason Wang, Li Zeng, Haofen Wang, Guilin Qi, Huajun Chen, and Ningyu Zhang. 2026. MobileMem: Learning from a Year of Mobile Experiences. arXiv preprint arXiv:2608.13606. arXiv:2608.13606 <https://arxiv.org/abs/2608.13606>
- [41] Yongheng Deng, Tianyuan Jiang, Zhenya Ma, Hao Wu, Yongjian Fu, Hao Pan, Sheng Yue, and Ju Ren. 2026. Accelerating Graph-Based RAG Retrieval via Locality-Aware Device-Cloud Collaboration. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*. Association for Computing Machinery, 12 pages. <https://doi.org/10.1145/3770855.3817674>
- [42] V. S. Dorbala and D. Manocha. 2026. MemCtrl: Using MLLMs as Active Memory Controllers in Embodied Agents. arXiv preprint arXiv:2601.20831. <https://arxiv.org/abs/2601.20831>
- [43] Y. Du, B. Wang, Y. He, B. Liang, B. Wang, Z. Li, L. Gui, J. Z. Pan, R. Xu, and K.-F. Wong. 2025. MemGuide: Intent-Driven Memory Selection for Goal-Oriented Multi-Session LLM Agents. arXiv preprint arXiv:2505.20231. <https://arxiv.org/abs/2505.20231>
- [44] R. A. El Haddad, Z. Wang, Y. Shin, R. Liu, Y. Wang, and C. Yu. 2025. AR Secretary Agent: Real-time Memory Augmentation via LLM-powered Augmented Reality Glasses. arXiv preprint arXiv:2505.11888. <https://arxiv.org/abs/2505.11888>
- [45] H. Fan, Y. Yuan, C. Hsia, C. Liang, and Z. Peng. 2026. Toward Personalized Proactive Agents on Smart Glasses: From Explicit Context Policies to Implicit User Differences. *Proceedings of the Extended Abstracts of the 2026 CHI Conference on Human Factors in Computing Systems*, 1-6. <https://doi.org/10.1145/3772363.3798509>
- [46] J. Fang, X. Deng, H. Xu, Z. Jiang, Y. Tang, Z. Xu, S. Deng, Y. Yao, M. Wang, S. Qiao, H. Chen, and N. Zhang. 2026. LightMem: Lightweight and Efficient Memory-Augmented Generation. *International Conference on Learning Representations*. <https://arxiv.org/abs/2510.18866>
- [47] Yichu Fang, Sitong Wei, Haozhe Hu, and Xiaoyu Shen. 2026. ReCache: Efficient KV Cache Reuse and Compression for Tool-Augmented LLM Agents. arXiv:2608.19662 [cs.CL] <https://arxiv.org/abs/2608.19662>
- [48] J. Gong, J. Ding, F. Meng, G. Chen, H. Chen, S. Zhao, H. Lu, and Y. Li. 2024. A Population-to-individual Tuning Framework for Adapting Pretrained LM to On-device User Intent Prediction. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 896-907. <https://doi.org/10.1145/3637528.3671984>
- [49] Anusha Madan Gopal, Aras Pirbadian, Kristofer D. Carlson, M Anthony Lewis, and Jonathan Tapson. 2026. Structured Memory for Edge Language Models: Persistent Context and Corpus Retrieval via O(1) SSM State Injection. arXiv preprint arXiv:2608.02560. arXiv:2608.02560 <https://arxiv.org/abs/2608.02560>
- [50] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, M. Martin, T. Nagarajan, I. Radosavovic, S. K. Ramakrishnan, F. Ryan, J. Sharma, M. Wray, M. Xu, E. Z. Xu, et al. 2022. Ego4D: Around the World in 3,000 Hours of Egocentric Video. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. <https://arxiv.org/abs/2110.07058>
- [51] Yuchen Guo, Junli Gong, Weicheng Wang, Hongmin Cai, Yiu ming Cheung, and Weifeng Su. 2026. PEAM: Parametric Embodied Agent Memory through Contrastive Internalization of Experience in Minecraft. arXiv:2605.27762 [cs.AI] <https://arxiv.org/abs/2605.27762>
- [52] G. Gupta, K. Yadav, Z. Kira, Y. Gal, and R. Aljundi. 2025. Memo: Training memory-efficient embodied agents with reinforcement learning. *Advances in Neural Information Processing Systems*, 38. https://proceedings.neurips.cc/paper_files/paper/2025/hash/96889893231d651898b0de42fdbec3a6-Abstract-Conference.html
- [53] Z. He, Y. Wang, C. Zhi, Y. Hu, T.-P. Chen, L. Yin, Z. Chen, T. A. Wu, S. Ouyang, Z. Wang, J. Pei, J. McAuley, Y. Choi, and A. Pentland. 2026. MemoryArena: Benchmarking Agent Memory in Interdependent Multi-Session Agentic Tasks. arXiv preprint arXiv:2602.16313. <https://arxiv.org/abs/2602.16313>
- [54] Peyman Hosseini, Ondrej Bohdal, Ahmed Alajrami, Andrea Maracani, Ignacio Castro, Matthew Purver, Mete Ozay, Savas Ozkan, and Taha Ceritli. 2026. DuoMem: Towards Capable On-Device Memory Agents via Dual-Space Distillation. arXiv:2606.29961 [cs.LG] <https://arxiv.org/abs/2606.29961>
- [55] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. 2022. LoRA: Low-rank adaptation of large language models. *International Conference on Learning Representations*. <https://openreview.net/forum?id=nZVeKeeFYf9>

- [56] M. Hu, T. Chen, Q. Chen, Y. Mu, W. Shao, and P. Luo. 2025. HiAgent: Hierarchical Working Memory Management for Solving Long-Horizon Agent Tasks with Large Language Model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 32779–32798). <https://doi.org/10.18653/v1/2025.acl-long.1575>
- [57] Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, Senjie Jin, Jiejun Tan, Yanbin Yin, Jiongnan Liu, Zeyu Zhang, Zhongxiang Sun, Yutao Zhu, Hao Sun, Boci Peng, Zhenrong Cheng, Xuanbo Fan, Jiaxin Guo, Xinlei Yu, Zhenhong Zhou, Zewen Hu, Jiahao Huo, Junhao Wang, Yuwei Niu, Yu Wang, Zhenfei Yin, Xiaobin Hu, Yue Liao, Qiankun Li, Kun Wang, Wangchunshu Zhou, Yixin Liu, Dawei Cheng, Qi Zhang, Tao Gui, Shirui Pan, Yan Zhang, Philip Torr, Zhicheng Dou, Ji-Rong Wen, Xuanjing Huang, Yu-Gang Jiang, and Shuicheng Yan. 2025. Memory in the Age of AI Agents. *arXiv preprint arXiv:2512.13564*. <https://arxiv.org/abs/2512.13564>
- [58] Y. Hu, Z. Long, J. Guo, X. Sui, X. Fu, W. Zhao, Y. Zhao, and B. Qin. 2026. OP-Bench: Benchmarking Over-Personalization for Memory-Augmented Personalized Conversational Agents. *arXiv preprint arXiv:2601.13722*. <https://arxiv.org/abs/2601.13722>
- [59] Yuanzhe Hu, Y. Wang, and J. McAuley. 2026. Evaluating memory in LLM agents via incremental multi-turn interactions. *International Conference on Learning Representations*. <https://openreview.net/forum?id=DT7JyQC3MR>
- [60] H. Jiang, D. Qiu, A. Stanesco, Y. Wang, H. B.-L. Duh, and F. Guan. 2025. PITAR: an LLM-powered Agent towards Intelligent and Accurate Manipulations in Extended Reality with Multimodal Interactions. *SIGGRAPH Asia 2025 XR*, 1-2. <https://doi.org/10.1145/3761667.3761944>
- [61] H. Jiang, X. Zhang, S. Garg, R. Arora, S.-Z. Kuo, J. Xu, A. Bansal, C. Brossman, Y. Liu, A. Colak, A. Aly, A. Kumar, and X. L. Dong. 2025. Memory-QA: Answering Recall Questions Based on Multimodal Memories. *arXiv preprint arXiv:2509.18436*. <https://arxiv.org/abs/2509.18436>
- [62] P. Jiang, F. Liu, J. Guo, C.-K. Wen, S. Jin, and J. Zhang. 2026. Intention-Aware Semantic Agent Communications for AI Glasses. *arXiv preprint arXiv:2604.23691*. <https://arxiv.org/abs/2604.23691>
- [63] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay. 2007. Evaluating the accuracy of implicit feedback from clicks and query reformulations in Web search. *ACM Transactions on Information Systems*, 25(2), 7. <https://doi.org/10.1145/1229179.1229181>
- [64] A. Jonelagadda, C. Hahn, H. Zheng, and S. Penachio. 2025. Mnemosyne: An unsupervised, human-inspired long-term memory architecture for edge-based LLMs. *arXiv preprint arXiv:2510.08601*. <https://arxiv.org/abs/2510.08601>
- [65] J. Kang, M. Ji, Z. Zhao, and T. Bai. 2025. Memory OS of AI Agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 25961–25970). <https://doi.org/10.18653/v1/2025.emnlp-main.1318>
- [66] D. Kaviani, A. E. Ozdarendeli, J. Zhu, Y. Ding, and R. A. Popa. 2026. Opal: Private Memory for Personal AI. *arXiv preprint arXiv:2604.02522*. <https://arxiv.org/abs/2604.02522>
- [67] D. Kelly and J. Teevan. 2003. Implicit feedback for inferring user preference. *ACM SIGIR Forum*, 37(2), 18-28. <https://doi.org/10.1145/959258.959260>
- [68] D. Khan, A. Kouyoumdjian, X. Liu, O. Mena, D. Engel, and I. Viola. 2026. ClickAIXR: On-Device Multimodal Vision-Language Interaction with Real-World Objects in Extended Reality. *arXiv preprint arXiv:2604.04905*. <https://arxiv.org/abs/2604.04905>
- [69] Minsoo Kim, Arnav Kundu, Han-Byul Kim, Richa Dixit, and Minsik Cho. 2026. EpiCache: Episodic KV Cache Management for Long-Term Conversation on Resource-Constrained Environments. In *International Conference on Machine Learning*. <https://icml.cc/virtual/2026/poster/65405>
- [70] Minsoo Kim, Kyuhong Shim, Jungwook Choi, and Simyung Chang. 2025. InfiniPot-V: Memory-Constrained KV Cache Compression for Streaming Video Understanding. In *Advances in Neural Information Processing Systems*, Vol. 38. <https://neurips.cc/virtual/2025/poster/116667>
- [71] Y. Kim, D. Jadeja, D. Pradhan, Y. Yang, and A. E. Kaufman. 2026. SpeechLess: Micro-utterance with Personalized Spatial Memory-aware Assistant in Everyday Augmented Reality. 2026 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), 217-227. <https://doi.org/10.1109/VR67842.2026.00044>
- [72] Johannes Kirmayr, Lukas Stappen, Phillip Schneider, Florian Matthes, and Elisabeth André. 2025. CarMem: Enhancing Long-Term Memory in LLM Voice Assistants through Category-Bounding. *arXiv preprint arXiv:2501.09645*. <https://arxiv.org/abs/2501.09645>
- [73] Abdulkadir Külçe, Alihan Esen, Çağla Fikir, Berke Kurt, Kuzey Arar, Gökhan Ercan, and Faik Boray Tek. 2026. ECHO: A Locally-Deployable Agentic Health Assistant with Temporal Memory, Safety Guardrails, and Speech Assessment. *arXiv preprint arXiv:2608.06110*. <https://arxiv.org/abs/2608.06110>
- [74] L. S. Kumar, Y. Ba, and R. Pan. 2026. MemArchitect: A Policy Driven Memory Governance Layer. *arXiv preprint arXiv:2603.18330*. <https://arxiv.org/abs/2603.18330>
- [75] T. Kwon, D. Choi, H. Kim, S. Kim, S. Moon, B.-W. Kwak, K.-H. Huang, and J. Yeo. 2026. Embodied Agents Meet Personalization: Investigating Challenges and Solutions Through the Lens of Memory Utilization. *International*

Conference on Learning Representations. <https://arxiv.org/abs/2505.16348>

- [76] Changmin Lee, Jaemin Kim, and Taesik Gong. 2026. From Volume to Value: Preference-Aligned Memory Construction for On-Device RAG. arXiv:2605.18271 [cs.CL] <https://arxiv.org/abs/2605.18271>
- [77] Juntae Lee, Jihwan Bang, Seunghan Yang, and Simyung Chang. 2025. CIFLEX: Contextual Instruction Flow for Sub-task Execution in Multi-Turn Interactions with a Single On-Device LLM. arXiv:2510.01239 [cs.CL] <https://arxiv.org/abs/2510.01239>
- [78] Rui Li, Zeyu Zhang, Xiaohe Bo, Zihang Tian, Xu Chen, Quanyu Dai, Zhenhua Dong, and Ruiming Tang. 2025. CAM: A Constructivist View of Agentic Memory for LLM-Based Reading Comprehension. arXiv:2510.05520 [cs.CL] <https://arxiv.org/abs/2510.05520>
- [79] W. Li, Z. Zhang, M. Zhang, G. Hong, M. Wen, X. You, and M. Yang. 2026. From Anchors to Supervision: Memory-Graph Guided Corpus-Free Unlearning for Large Language Models. arXiv preprint arXiv:2604.13777. <https://arxiv.org/abs/2604.13777>
- [80] X. Li, Y. Zhang, F. Zhou, P. Zhang, and X. Ji. 2026. Efficient on-device continuous learning enabling evolutionary edge intelligence. *Neurocomputing*, Article 132515. <https://doi.org/10.1016/j.neucom.2025.132515>
- [81] Y. Li, S. Banerjee, and T. Che. 2026. EMBER: Efficient Memory via Budgeted Evidence Retention for Long-Horizon Agents. arXiv preprint arXiv:2606.05894. <https://arxiv.org/abs/2606.05894>
- [82] Y. Li, C. Zhang, W. Jiang, W. Yang, B. Fu, P. Cheng, X. Chen, L. Chen, and Y. Wei. 2024. AppAgent v2: Advanced Agent for Flexible Mobile Interactions. arXiv preprint arXiv:2408.11824. <https://arxiv.org/abs/2408.11824>
- [83] K. Liang, J. Kruk, S. Qian, X. Yang, S. Bi, Y. Yao, S. Nie, M. Zhang, L. Liu, J. F. Fisac, S. Zhou, and S. Hosseini. 2026. Learning Personalized Agents from Human Feedback. arXiv preprint arXiv:2602.16173. <https://arxiv.org/abs/2602.16173>
- [84] C. Liu, L. Chen, H. Zhou, X. Zhang, Q. Kong, P. Tong, W. Wang, X. Yu, S. Hoi, and Y. Wang. 2026. What Memory Do GUI Agents Really Need? From Passive Records to Active Task-Driving States. arXiv preprint arXiv:2606.31612. <https://arxiv.org/abs/2606.31612>
- [85] C. Liu, Y. Yang, S. X. Pu, Y. Liu, L. Long, Y. Guo, N. Chen, Z. Weng, E. Kochkina, S. Kaur, C. Smiley, X. Liu, J. Zou, S. Liu, Y. Bu, S. Peng, and X. E. Wang. 2026. WorldMemArena: Evaluating Multimodal Agent Memory Through Action-World Interaction. arXiv preprint arXiv:2605.29341. <https://arxiv.org/abs/2605.29341>
- [86] G. Liu and S. Gabriel. 2026. PM-Bench: Evaluating Prospective Memory in LLM Agents. *Conference on Language Modeling*. <https://arxiv.org/abs/2607.12385>
- [87] G. Liu, P. Zhao, Y. Liang, Q. Luo, S. Tang, Y. Chai, W. Lin, H. Xiao, W. Wang, S. Chen, Z. Lu, G. Wu, H. Wang, L. Liu, and Y. Liu. 2026. MemGUI-Bench: Benchmarking Memory of Mobile GUI Agents in Dynamic Environments. arXiv preprint arXiv:2602.06075. <https://arxiv.org/abs/2602.06075>
- [88] Mugeng Liu, Shuoqi Li, Qi Yang, and Yun Ma. 2026. WAB: Overcoming Memory, Network, and Security Walls in Native Agentic Browsers with WebAssembly. In *Companion Proceedings of the ACM Web Conference 2026*. Association for Computing Machinery, 4 pages. <https://doi.org/10.1145/3774905.3795093>
- [89] Siyuan Liu and Huangxun Chen. 2026. MemAura: Structured Context Memory for Personalized LLM Reasoning in Smart Environments. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 10, 2, Article 55 (2026), 43 pages. <https://doi.org/10.1145/3810196>
- [90] S. Liu, J. Zhu, L. Shu, J. Lin, Y. Chen, H. Zhang, C. Zhang, D. Xu, J. Li, B. Tang, Z. Li, F. Xiong, E. Chen, and T. Xu. 2026. PERMA: Benchmarking Personalized Memory Agents via Event-Driven Preference and Realistic Task Environments. arXiv preprint arXiv:2603.23231. <https://arxiv.org/abs/2603.23231>
- [91] Xinxin Liu, Jiaxin Li, Zibo Wang, Yun Ji, Zhangqi Zhu, Qing Hu, Zhibin Wang, Rong Gu, Sheng Zhong, and Chen Tian. 2026. LOCAL: Enabling Learning On-device Contiguously for Agent LLMs. arXiv preprint arXiv:2608.15241. arXiv:2608.15241 <https://arxiv.org/abs/2608.15241>
- [92] Z. Liu, C. Zhang, X. Zhao, Y. Feng, B. Bai, D. Feng, E. Feng, Y. Xia, and H. Chen. 2025. Beyond Training: Enabling Self-Evolution of Agents with MOBIMEM. arXiv preprint arXiv:2512.15784. <https://arxiv.org/abs/2512.15784>
- [93] L. Long, Y. He, W. Ye, Y. Pan, Y. Lin, H. Li, J. Zhao, and W. Li. 2025. Seeing, Listening, Remembering, and Reasoning: A Multimodal Agent with Long-Term Memory. arXiv preprint arXiv:2508.09736. <https://arxiv.org/abs/2508.09736>
- [94] Gowen Loo, Chang Liu, Qinghong Yin, Xiang Chen, Jiawei Chen, Jingyuan Zhang, and Yu Tian. 2025. MobileRAG: Enhancing Mobile Agent with Retrieval-Augmented Generation. arXiv preprint arXiv:2509.03891. arXiv:2509.03891 <https://arxiv.org/abs/2509.03891>
- [95] Zhenyan Lu, Daliang Xu, Dongqi Cai, Zexi Li, Wei Liu, Jian Luan, Fangming Liu, Shangguang Wang, and Mengwei Xu. 2026. MobiEdit: Resource-efficient Knowledge Editing for Personalized On-device LLMs. In *International Conference on Learning Representations*. https://proceedings.iclr.cc/paper_files/paper/2026/hash/2b35d186908df6fa14a51ba1cae9dc4e-Abstract-Conference.html
- [96] Y. Lyu, G. Chen, R. Shao, W. Guan, and L. Nie. 2026. PersonalAlign: Hierarchical Implicit Intent Alignment for Personalized GUI Agent with Long-Term User-Centric Records. In *Proceedings of the 64th Annual Meeting of the*

- Association for Computational Linguistics (Volume 1: Long Papers) (pp. 36074–36089). <https://doi.org/10.18653/v1/2026.acl-long.1669>
- [97] A. Maharana, D.-H. Lee, S. Tulyakov, M. Bansal, F. Barbieri, and Y. Fang. 2024. Evaluating Very Long-Term Conversational Memory of LLM Agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 13851–13870). <https://doi.org/10.18653/v1/2024.acl-long.747>
 - [98] M. M. Maheri, S. Cotterill, A. Davidson, and H. Haddadi. 2026. ZK-APEX: Zero-Knowledge Approximate Personalized Unlearning with Executable Proofs. *Proceedings of the 9th Conference on Machine Learning and Systems*. <https://arxiv.org/abs/2512.09953>
 - [99] Ripon Chandra Malo and Tong Qiu. 2026. PROJECTMEM: A Local-First, Event-Sourced Memory and Judgment Layer for AI Coding Agents. *arXiv preprint arXiv:2606.12329*. [arXiv:2606.12329](https://arxiv.org/abs/2606.12329) <https://arxiv.org/abs/2606.12329>
 - [100] Natasha Maniar, Samantha Chan, Wazeer Zulfikar, Scott Ren, Christine Xu, and Pattie Maes. 2025. MemPal: Leveraging Multimodal AI and LLMs for Voice-Activated Object Retrieval in Homes of Older Adults. In *Proceedings of the 30th International Conference on Intelligent User Interfaces*. <https://doi.org/10.1145/3708359.3712151>
 - [101] Wenyu Mao, Haosong Tan, Shuchang Liu, Haoyang Liu, Yifan Xu, Huaxiang Ji, and Xiang Wang. 2026. Bi-Mem: Bidirectional Construction of Hierarchical Memory for Personalized LLMs via Inductive-Reflective Agents. *arXiv:2601.06490* [cs.MA] <https://arxiv.org/abs/2601.06490>
 - [102] J. Mei, J. Chen, G. Yang, X. Hou, M. Li, and B. Byrne. 2026. According to Me: Long-Term Personalized Referential Memory QA. *arXiv preprint arXiv:2603.01990*. <https://arxiv.org/abs/2603.01990>
 - [103] M. A. Merrill, A. Paruchuri, N. Rezaei, G. Kovacs, J. Perez, Y. Liu, E. Schenck, N. Hammerquist, J. Sunshine, S. Tailor, K. Ayush, H.-W. Su, Q. He, C. Y. McLean, M. Malhotra, S. Patel, J. Zhan, T. Althoff, D. McDuff, and X. Liu. 2026. Transforming wearable data into personal health insights using large language model agents. *Nature Communications*, 17(1), 1143. <https://doi.org/10.1038/s41467-025-67922-y>
 - [104] H. Mohammed, H. Yin, and S. C. Boyapati. 2025. Context adaptive memory-efficient LLM inference for edge multi-agent systems: Extended abstract. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems* (pp. 2678–2680). <https://dl.acm.org/doi/10.5555/3709347.3743976>
 - [105] V. Mollyn, K. Ahuja, D. Verma, C. Harrison, and M. Goel. 2022. SAMoSA: Sensing activities with motion and subsampled audio. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(3), 1-19. <https://doi.org/10.1145/3550284>
 - [106] Reina Mun, Zishen Wan, and Vijay Janapa Reddi. 2026. Affective Agent: On-Device Personalized Intervention Reasoning for Wearable Systems. *IEEE Internet Computing* (2026). <https://doi.org/10.1109/MIC.2026.3732091>
 - [107] H. Nie, X. Liu, Y. Bai, Y. Wang, Y. Liu, Q. Yao, and Z. Wang. 2026. PSPA-Bench: A Personalized Benchmark for Smartphone GUI Agent. *arXiv preprint arXiv:2603.29318*. <https://arxiv.org/abs/2603.29318>
 - [108] Koji Nishikawa and Makoto P. Kato. 2026. H-MAPS: Hierarchical Memory-Augmented Proactive Search Assistant for Scientific Literature. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*. <https://doi.org/10.1145/3805712.3808378>
 - [109] Zaifeng Pan, Ajikumar Patel, Zhengding Hu, Yipeng Shen, Yue Guan, Wan-Lu Li, Lianhui Qin, Yida Wang, and Yufei Ding. 2025. KVFlow: Efficient Prefix Caching for Accelerating LLM-Based Multi-Agent Workflows. *arXiv:2507.07400* [cs.DC] <https://arxiv.org/abs/2507.07400>
 - [110] Joon Sung Park, Joseph O’Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative Agents: Interactive Simulacra of Human Behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, NY, USA, 1–22. <https://arxiv.org/abs/2304.03442>
 - [111] T. Park, G. Lee, and M.-S. Kim. 2025. MobileRAG: A Fast, Memory-Efficient, and Energy-Efficient Method for On-Device RAG. *arXiv preprint arXiv:2507.01079*. <https://arxiv.org/abs/2507.01079>
 - [112] A. Paruchuri, S. Hersek, L. Aggarwal, Q. Yang, X. Liu, A. Kulshrestha, A. Colaco, H. Fuchs, and I. Chatterjee. 2025. EgoTrigger: Toward Audio-Driven Image Capture for Human Memory Enhancement in All-Day Energy-Efficient Smart Glasses. *IEEE Transactions on Visualization and Computer Graphics* 31, 11 (2025), 9720–9729. <https://doi.org/10.1109/TVCG.2025.3616866>
 - [113] Ishan Patel and Ishan Joshi. 2026. PolyKV: Shared Asymmetric TurboQuant-Compressed KV Memory for Multi-Agent LLMs. <https://doi.org/10.5281/ZENODO.19686729>
 - [114] K. Pu, T. Zhang, N. Sendhilnathan, S. Freitag, R. Sodhi, and T. R. Jonker. 2025. ProMemAssist: Exploring Timely Proactive Assistance Through Working Memory Modeling in Multi-Modal Wearable Devices. *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*, 1-19. <https://doi.org/10.1145/3746059.3747770>
 - [115] PUBG Ally Team. 2026. PUBG Ally: A Conversational Embodied Agent as an AI Teammate. *arXiv preprint arXiv:2609.29837*. [arXiv:2609.29837](https://arxiv.org/abs/2609.29837) <https://arxiv.org/abs/2609.29837>
 - [116] Zerui Qin, Sheng Yue, Xingyuan Hua, Yongjian Fu, and Ju Ren. 2026. Executable Agentic Memory for GUI Agent. In *International Conference on Machine Learning*. <https://icml.cc/virtual/2026/poster/65931>

- [117] J. Rachapudi, P. Singh, R. Vatsi, P. Hambarde, and A. Shukla. 2026. RePAIR: Interactive Machine Unlearning through Prompt-Aware Model Repair. arXiv preprint arXiv:2604.12820. <https://arxiv.org/abs/2604.12820>
- [118] Aditya Karnam Gururaj Rao and Arjun Jaggi. 2026. BudgetBench: A Budget-Tiered Protocol and Pilot Harness for Memory Strategy Evaluation in Local Large Language Model Agents. arXiv preprint arXiv:2609.13149. arXiv:2609.13149 <https://arxiv.org/abs/2609.13149>
- [119] P. Rasmussen, P. Paliychuk, T. Beauvais, J. Ryan, and D. Chalef. 2025. Zep: A Temporal Knowledge Graph Architecture for Agent Memory. arXiv preprint arXiv:2501.13956. <https://arxiv.org/abs/2501.13956>
- [120] C. Rawles, S. Clinckemallie, Y. Chang, J. Waltz, G. Lau, M. Fair, A. Li, W. Bishop, W. Li, F. Campbell-Ajala, D. Toyama, R. Berry, D. Tyamagundlu, T. Lillicrap, and O. Riva. 2025. AndroidWorld: A Dynamic Benchmarking Environment for Autonomous Agents. International Conference on Learning Representations. <https://arxiv.org/abs/2405.14573>
- [121] A. Rege, A. Sadhu, Y. Li, K. Li, R. K. Vinayak, Y. Chai, Y. J. Lee, and H. J. Kim. 2026. Agentic Very Long Video Understanding. arXiv preprint arXiv:2601.18157. <https://arxiv.org/abs/2601.18157>
- [122] Tao Ren, Weiyao Luo, Hui Yang, Rongzhi Zhu, Xiang Huang, Yuchuan Wu, Bingxue Chou, Jieping Ye, Jiafeng Liang, Yongbin Li, and Yijie Peng. 2026. Scaling Self-Evolving Agents via Parametric Memory. arXiv:2606.04536 [cs.AI] <https://arxiv.org/abs/2606.04536>
- [123] J. Saad-Falcon, A. Narayan, R. Manihani, T. Bhatthal, H. Shandilya, H. O. Akengin, G. Bo, A. Park, M. Hart, C. Costello, C. Li, C. Ré, and A. Mirhoseini. 2026. OpenJarvis: Personal AI, On Personal Devices. arXiv preprint arXiv:2605.17172. <https://arxiv.org/abs/2605.17172>
- [124] Alaa Saleh, Sasu Tarkoma, Praveen Kumar Donta, Anders Lindgren, Naser Hossein Motlagh, Schahram Dustdar, Susanna Pirttikangas, and Lauri Lovén. 2025. UserCentrix: An Agentic Memory-augmented AI Framework for Smart Spaces. arXiv preprint arXiv:2505.00472. arXiv:2505.00472 <https://arxiv.org/abs/2505.00472>
- [125] Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2024. LaMP: When Large Language Models Meet Personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 7370–7392. <https://doi.org/10.18653/v1/2024.acl-long.399>
- [126] Baocai Shan, Yuzhuang Xu, and Wanxiang Che. 2026. HUOZIIME: An On-Device LLM-enhanced Input Method for Deep Personalization. arXiv preprint arXiv:2604.14159. arXiv:2604.14159 <https://arxiv.org/abs/2604.14159>
- [127] Y. Shen, K. Li, W. Zhou, and S. Hu. 2026. Mem2ActBench: A Benchmark for Evaluating Long-Term Memory Utilization in Task-Oriented Autonomous Agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 8173–8190). <https://doi.org/10.18653/v1/2026.acl-long.370>
- [128] D. Shenaj, O. Bohdal, T. Ceritli, M. Ozay, P. Zanuttigh, and U. Michieli. 2026. K-Merge: Online Continual Merging of Adapters for On-device Large Language Models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 3013–3029). <https://doi.org/10.18653/v1/2026.acl-long.137>
- [129] J. Shi, H. Zhang, Y. Zhang, T. L. Lam, L. Zhang, H. Huang, and Y. Gao. 2026. VLONS: A Vision-and-Language On-Device Navigation System With Multimodal Fusion and Modular Framework. *IEEE Transactions on Consumer Electronics*, 72(1), 1171–1182. <https://doi.org/10.1109/TCE.2025.3638139>
- [130] Yunxiao Shi, Wujiang Xu, Zeqi Zhang, Xing Zi, Qiang Wu, and Min Xu. 2025. PersonaX: A Recommendation Agent Oriented User Modeling Framework for Long Behavior Sequence. arXiv:2503.02398 [cs.IR] <https://arxiv.org/abs/2503.02398>
- [131] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems*, Vol. 36. 8634–8652. https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd88628510e90-Abstract-Conference.html
- [132] Y. P. Shkolnikov. 2026. Agent Memory Below the Prompt: Persistent Q4 KV Cache for Multi-Agent LLM Inference on Edge Devices. arXiv preprint arXiv:2603.04428. <https://arxiv.org/abs/2603.04428>
- [133] Y. Shu, S. P. Jonnalagedda, X. Gao, B. J. Gutiérrez, W. Qi, K. Das, H. Sun, and Y. Su. 2026. REMem: Reasoning with Episodic Memory in Language Agent. International Conference on Learning Representations. <https://arxiv.org/abs/2602.13530>
- [134] Zhihao Shu, Md Musfiqur Rahman Sanim, Jie Hu, Kun Yuan, Minghai Qin, Gagan Agrawal, and Wei Niu. 2026. TierKV: Long-Context On-Device LLMs via Predictive Multi-Tier KV Caching. arXiv:2609.21172 [cs.LG] <https://arxiv.org/abs/2609.21172>
- [135] B. Simsek. 2026. When to Forget: A Memory Governance Primitive. arXiv preprint arXiv:2604.12007. <https://arxiv.org/abs/2604.12007>
- [136] I. Song and K. Lee. 2026. BitLoRA: Quantization-Compatible Adapter Tuning for 1.58-bit LLM in Federated On-Device AI-Agent. Expert Systems with Applications, 311, 131397. <https://doi.org/10.1016/j.eswa.2026.131397>
- [137] Larry R. Squire and Adam J. O. Dede. 2015. Conscious and Unconscious Memory Systems. *Cold Spring Harbor Perspectives in Biology* 7, 3 (2015), a021667. <https://doi.org/10.1101/cshperspect.a021667>

- [138] Felix Stocker. 2026. Towards Persistent Case-Based Memory for Autonomous Data Science: A CBR-Augmented R&D-Agent with a Locally Deployable Small Language Model. arXiv preprint arXiv:2606.05250. arXiv:2606.05250 <https://arxiv.org/abs/2606.05250>
- [139] Theodore R. Summers, Shunyu Yao, Karthik Narasimhan, and Thomas L. Griffiths. 2024. Cognitive Architectures for Language Agents. *Transactions on Machine Learning Research* (2024). <https://arxiv.org/abs/2309.02427>
- [140] Lizheng Sun. 2026. MemX: A Local-First Long-Term Memory System for AI Assistants. arXiv preprint arXiv:2603.16171. arXiv:2603.16171 <https://arxiv.org/abs/2603.16171>
- [141] L. Sun, J. Zhang, S. Wang, and Z. Wei. 2026. MAGNET: Towards Adaptive GUI Agents with Memory-Driven Knowledge Evolution. arXiv preprint arXiv:2601.19199. <https://arxiv.org/abs/2601.19199>
- [142] S. Sun, K. Han, Y. Huang, W. Cai, and J. Song. 2026. EgoGraph: Temporal Knowledge Graph for Egocentric Video Understanding. arXiv preprint arXiv:2602.23709. <https://arxiv.org/abs/2602.23709>
- [143] H. Tan, Z. Zhang, C. Ma, X. Chen, Q. Dai, and Z. Dong. 2025. MemBench: Towards More Comprehensive Evaluation on the Memory of LLM-based Agents. In Findings of the Association for Computational Linguistics: ACL 2025 (pp. 19336–19352). <https://doi.org/10.18653/v1/2025.findings-acl.989>
- [144] Z. Tan, J. Yan, I.-H. Hsu, R. Han, Z. Wang, L. Le, Y. Song, Y. Chen, H. Palangi, G. Lee, A. R. Iyer, T. Chen, H. Liu, C.-Y. Lee, and T. Pfister. 2025. In Prospect and Retrospect: Reflective Memory Management for Long-term Personalized Dialogue Agents. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 8416–8439). <https://doi.org/10.18653/v1/2025.acl-long.413>
- [145] Y. Tang, J. Zhang, X. Qin, J. Yu, M. Qiu, G. Gou, G. Xiong, Q. Lin, S. Rajmohan, D. Zhang, and Q. Wu. 2026. EgoMemory: Memory-Augmented Personalized Retrieval for Long-Context Egocentric Video. In Findings of the Association for Computational Linguistics: ACL 2026 (pp. 7317–7349). <https://doi.org/10.18653/v1/2026.findings-acl.362>
- [146] R. Tiwari, H. Xi, A. Tomar, C. Hooper, S. Kim, M. Horton, M. Najibi, M. W. Mahoney, K. Keutzer, and A. Gholami. 2025. QuantSpec: Self-Speculative Decoding with Hierarchical Quantized KV Cache. arXiv preprint arXiv:2502.10424. <https://arxiv.org/abs/2502.10424>
- [147] M. N. Uddin, K. Shubham, E. Blanco, C. Baral, and G. Wang. 2026. From Recall to Forgetting: Benchmarking Long-Term Memory for Personalized Agents. In Findings of the Association for Computational Linguistics: ACL 2026 (pp. 26814–26841). <https://doi.org/10.18653/v1/2026.findings-acl.1337>
- [148] Yash Vekaria, Aurelio Loris Canino, Jonathan Levitsky, Alex Ciechonski, Patricia Callejo, Anna Maria Mandalari, and Zubair Shafiq. 2025. Big Help or Big Brother? Auditing Tracking, Profiling, and Personalization in Generative AI Assistants. In *34th USENIX Security Symposium (USENIX Security 25)*. USENIX Association, 8115–8134. <https://www.usenix.org/conference/usenixsecurity25/presentation/vekaria>
- [149] S. Vijayvargiya and R. Lokesh. 2025. Efficient On-Device Agents via Adaptive Context Management. arXiv preprint arXiv:2511.03728. <https://arxiv.org/abs/2511.03728>
- [150] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar. 2024. Voyager: An Open-Ended Embodied Agent with Large Language Models. *Transactions on Machine Learning Research* (2024). <https://openreview.net/forum?id=ehfRiF0R3a>
- [151] Huili Wang, Yuanhong Huang, Zhiyang Hu, Qing Li, Pingyi Fan, Yongfeng Huang, Shangguang Wang, and Tao Qi. 2026. Efficient and Effective Personalized In-Context Learning for On-Device Large Model Services. *IEEE Transactions on Services Computing* 19, 4 (2026), 2788–2800. <https://doi.org/10.1109/TSC.2026.3697569>
- [152] Heng Wang, Yifei Li, Lingling Zhang, Pengyu Li, Xinyu Che, Xinyu Zhang, and Zesheng Yang. 2026. QUMem: Personalized Memory for Query-Conditioned User-State Inference in LLM Agents. arXiv:2608.16168 [cs.CL] <https://arxiv.org/abs/2608.16168>
- [153] J. Wang, H. Xu, H. Jia, X. Zhang, M. Yan, W. Shen, J. Zhang, F. Huang, and J. Sang. 2024. Mobile-Agent-v2: Mobile Device Operation Assistant with Effective Navigation via Multi-Agent Collaboration. arXiv preprint arXiv:2406.01014. <https://arxiv.org/abs/2406.01014>
- [154] J. Wang, H. Xu, X. Zhang, Z. Zhu, M. Yan, J. Ye, and J. Sang. 2026. STAMP: Training Explicit Memory for Mobile GUI Agents in Controllable and Scalable Virtual Environments. arXiv preprint arXiv:2605.29324. <https://arxiv.org/abs/2605.29324>
- [155] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science* 18, 6, Article 186345 (2024). <https://doi.org/10.1007/s11704-024-40231-1>
- [156] Piaohong Wang, Motong Tian, Jiaxian Li, Yuan Liang, Yuqing Wang, Qianben Chen, Tiannan Wang, Zhicong Lu, Jiawei Ma, Yuchen Eleanor Jiang, and Wangchunshu Zhou. 2025. O-Mem: Omni Memory System for Personalized, Long Horizon, Self-Evolving Agents. arXiv:2511.13593 [cs.CL] <https://arxiv.org/abs/2511.13593>
- [157] S. Wang, C. Liu, G. Loo, L. Zheng, K. Wei, X. Zeng, J. Zhang, and Y. Tian. 2026. Me-Agent: A Personalized Mobile Agent with Two-Level User Habit Learning for Enhanced Interaction. arXiv preprint arXiv:2601.20162. <https://arxiv.org/abs/2601.20162>

- [158] T. Wang, M. Huang, F. Li, L. Chen, J. Zhang, and J. Ren. 2025. DynaKV: Enabling Accurate and Efficient Long-Sequence LLM Decoding on Smartphones. arXiv preprint arXiv:2511.07427. <https://arxiv.org/abs/2511.07427>
- [159] X. Wang, Z. Cui, H. Li, Y. Zeng, C. Wang, R. Song, Y. Chen, K. Shao, Q. Zhang, J. Liu, S. Ren, S. Hu, and Z. Wang. 2025. PerPilot: Personalizing VLM-based Mobile Agents via Memory and Exploration. arXiv preprint arXiv:2508.18040. <https://arxiv.org/abs/2508.18040>
- [160] Xubin Wang, Zhiqing Tang, Jianxiong Guo, Tianhui Meng, Chenhao Wang, Tian Wang, and Weijia Jia. 2025. Empowering Edge Intelligence: A Comprehensive Survey on On-Device AI Models. *Comput. Surveys* 57, 9 (2025), 1–39. <https://doi.org/10.1145/3724420>
- [161] X. Wang, X. Zhao, P. Rodriguez, D. S. Sachan, B. Oguz, S. Moon, S.-W. Li, G. Ghosh, X. Dong, and W.-T. Yih. 2026. S-EMBER: A Large-Scale Benchmark for Streaming Egocentric Memory Retrieval. arXiv preprint arXiv:2607.02689. <https://arxiv.org/abs/2607.02689>
- [162] Y. Wang and X. Chen. 2025. MIRIX: Multi-Agent Memory System for LLM-Based Agents. arXiv preprint arXiv:2507.07957. <https://arxiv.org/abs/2507.07957>
- [163] Y. Wang, Z. Li, S. Liu, Y. Wu, Z. Mao, Y. Zhao, X. Yan, Z. Xu, Y. Zhou, I. Stoica, S. Min, M. Zaharia, and J. E. Gonzalez. 2025. LEANN: A Low-Storage Vector Index. arXiv preprint arXiv:2506.08276. <https://arxiv.org/abs/2506.08276>
- [164] Y. Wang, Y. Xu, X. Li, J. Hong, Y. Wang, C. W. Chen, and W. Zhu. 2026. EgoSelf: From Memory to Personalized Egocentric Assistant. arXiv preprint arXiv:2604.19564. <https://arxiv.org/abs/2604.19564>
- [165] Ziyi Wang, Haonan Jin, Zian Wang, Wendong Wang, and Lanshan Zhang. 2026. LightWM: Training-Free Hierarchical Working Memory for Small Language Model Agents. In *Proceedings of the 43rd International Conference on Machine Learning*. PMLR. ICML 2026.
- [166] Z. Wang, Z. Li, Z. Jiang, D. Tu, and W. Shi. 2024. Crafting Personalized Agents through Retrieval-Augmented Generation on Editable Memory Graphs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing* (pp. 4891–4906). <https://doi.org/10.18653/v1/2024.emnlp-main.281>
- [167] Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. 2025. Agent Workflow Memory. In *Proceedings of the 42nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 267)*. PMLR, 63897–63911. <https://proceedings.mlr.press/v267/wang25bx.html>
- [168] B. Wu, Z. Ding, J. Huang, and Y. Zhao. 2026. Forget to Improve: On-Device LLM-Agent Continual Learning via Budget-Curated Memory. arXiv preprint arXiv:2606.25115. <https://arxiv.org/abs/2606.25115>
- [169] D. Wu, Z. Ji, A. Kawatkar, B. Kwan, J.-C. Gu, N. Peng, and K.-W. Chang. 2026. LongMemEval-V2: Evaluating Long-Term Agent Memory Toward Experienced Colleagues. arXiv preprint arXiv:2605.12493. <https://arxiv.org/abs/2605.12493>
- [170] D. Wu, H. Wang, W. Yu, Y. Zhang, K.-W. Chang, and D. Yu. 2025. LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory. *International Conference on Learning Representations*. <https://arxiv.org/abs/2410.10813>
- [171] J. Wu, M. Hu, J. Zhu, J. Wang, and Y. Jin. 2026. Agent-Memory Protocol: A privacy-focused protocol for LLM agents and user memory interaction. In *Proceedings of the Second AAAI Bridge Program on AI for Medicine and Healthcare* (PMLR Vol. 317). <https://proceedings.mlr.press/v317/wu26a.html>
- [172] W. Wu, K. Zhou, R. Yuan, V. Yu, S. Wang, Z. Hu, and B. Huang. 2025. Auto-scaling Continuous Memory for GUI Agent. arXiv preprint arXiv:2510.09038. <https://arxiv.org/abs/2510.09038>
- [173] Z. Wu, H. Huang, Y. Yang, Y. Song, X. Lou, W. Liu, W. Zhang, J. Wang, and Z. Zhang. 2025. Quick on the Uptake: Eliciting Implicit Intents from Human Demonstrations for Personalized Mobile-Use Agents. arXiv preprint arXiv:2508.08645. <https://arxiv.org/abs/2508.08645>
- [174] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huang, Qi Zhang, and Tao Gui. 2025. The rise and potential of large language model based agents: a survey. *Science China Information Sciences* 68, 2, Article 121101 (2025). <https://doi.org/10.1007/s11432-024-4222-0>
- [175] X. Xia, Z. Wang, R. Sun, B. Liu, I. Khalil, and M. Xue. 2024. Edge Unlearning is Not "on Edge"! An Adaptive Exact Unlearning System on Resource-Constrained Devices. arXiv preprint arXiv:2410.10128. <https://arxiv.org/abs/2410.10128>
- [176] H. Xiao, G. Wang, H. Wang, S. Liu, Y. Chai, Y. Pan, Y. Zhou, X. Chen, Y. Wen, and H. Li. 2026. UI-Mem: Self-Evolving Experience Memory for Online Reinforcement Learning in Mobile GUI Agents. arXiv preprint arXiv:2602.05832. <https://arxiv.org/abs/2602.05832>
- [177] Buqiang Xu, Yijun Chen, Jizhan Fang, Ruobin Zhong, Yunzhi Yao, Yuqi Zhu, Lun Du, and Shumin Deng. 2026. StructMem: Structured Memory for Long-Horizon Behavior in LLMs. arXiv:2604.21748 [cs.CL] <https://arxiv.org/abs/2604.21748>
- [178] Haichuan Xu, David Oygenblik, Runze Zhang, Mingxuan Yao, Muhammad Ibrahim, and Brendan Saltaformaggio. 2026. Recovering and Rehosting Mobile Local LLM Conversations and Contexts via Memory Forensics. In *2026 IEEE*

- Symposium on Security and Privacy.* <https://doi.org/10.1109/SP63933.2026.00252>
- [179] W. Xu, Z. Liang, K. Mei, H. Gao, J. Tan, and Y. Zhang. 2025. A-Mem: Agentic memory for LLM agents. *Advances in Neural Information Processing Systems*, 38. https://proceedings.neurips.cc/paper_files/paper/2025/hash/19909c36f51abc4856b4560aff3d36d6-Abstract-Conference.html
 - [180] S. Yan, X. Yang, Z. Huang, E. Nie, Z. Ding, Z. Li, X. Ma, J. Bi, K. Kersting, J. Z. Pan, H. Schuetze, V. Tresp, and Y. Ma. 2026. Memory-R1: Enhancing Large Language Model Agents to Manage and Utilize Memories via Reinforcement Learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 12805–12825). <https://doi.org/10.18653/v1/2026.acl-long.583>
 - [181] B. Yang, L. Xu, L. Zeng, Y. Guo, S. Jiang, W. Lu, K. Liu, Y. Li, X. Jiang, G. Xing, and Z. Yan. 2026. ProAgent: Harnessing On-Demand Sensory Contexts for Proactive LLM Agent Systems in the Wild. *arXiv preprint arXiv:2512.06721*. <https://arxiv.org/abs/2512.06721>
 - [182] B. Yang, L. Xu, L. Zeng, K. Liu, S. Jiang, W. Lu, H. Chen, X. Jiang, G. Xing, and Z. Yan. 2025. ContextAgent: Context-Aware Proactive LLM Agents with Open-World Sensory Perceptions. *Advances in Neural Information Processing Systems*, 38. <https://arxiv.org/abs/2505.14668>
 - [183] J. Yang, S. Liu, H. Guo, Y. Dong, X. Zhang, S. Zhang, P. Wang, Z. Zhou, B. Xie, Z. Wang, B. Ouyang, Z. Lin, M. Cominelli, Z. Cai, Y. Zhang, P. Zhang, F. Hong, J. Widmer, F. Gringoli, et al. 2025. EgoLife: Towards Egocentric Life Assistant. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 28885–28900. <https://arxiv.org/abs/2503.03803>
 - [184] S. Yang, J. Wu, D. F. Wong, and D. Wang. 2026. SelfMem: Self-Optimizing Memory for AI Agents. *arXiv preprint arXiv:2607.03726*. <https://arxiv.org/abs/2607.03726>
 - [185] Z. Yang, S. Zheng, T. Xie, T. Xu, B. Yu, F. Wang, J. Tang, S. Liu, and M. Li. 2025. EfficientNav: Towards on-device object-goal navigation with navigation map caching and retrieval. *Advances in Neural Information Processing Systems*, 38. https://proceedings.neurips.cc/paper_files/paper/2025/hash/067437c6d5d0369b6d09200bef89715b-Abstract-Conference.html
 - [186] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=WE_vluYUL-X
 - [187] D. Ye, T. Zhu, C. Zhu, F. He, Q. He, S. Wang, B. Liu, and W. Zhou. 2026. Secure Forgetting: A Framework for Privacy-Driven Unlearning in Large Language Model (LLM)-Based Agents. *arXiv preprint arXiv:2604.00430*. <https://arxiv.org/abs/2604.00430>
 - [188] Hancheng Ye, Zhengqi Gao, Mingyuan Ma, Qinsi Wang, Yuzhe Fu, Ming-Yu Chung, Yueqian Lin, Zhijian Liu, Jianyi Zhang, Danyang Zhuo, and Yiran Chen. 2025. KVCOMM: Online Cross-context KV-cache Communication for Efficient LLM-based Multi-agent Systems. *arXiv:2510.12872 [cs.MA]* <https://arxiv.org/abs/2510.12872>
 - [189] S. Ye, B. Ouyang, T. Qian, L. Zeng, M. Yuan, X. Chu, W. Hong, and X. Chen. 2026. Venus: An Efficient Edge Memory-and-Retrieval System for VLM-based Online Video Understanding. *IEEE INFOCOM 2026*. <https://arxiv.org/abs/2512.07344>
 - [190] H. Yin, Z. Wen, J. Cao, B. Yuan, and R. Yang. 2026. FOCAL: Filtered On-device Continuous Activity Logging for Efficient Personal Desktop Summarization. *arXiv preprint arXiv:2604.19541*. <https://arxiv.org/abs/2604.19541>
 - [191] Y. Yu, L. Yao, Y. Xie, Q. Tan, J. Feng, Y. Li, and L. Wu. 2026. Agentic Memory: Learning Unified Long-Term and Short-Term Memory Management for Large Language Model Agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 21457–21483). <https://doi.org/10.18653/v1/2026.acl-long.981>
 - [192] M. Yuan, H. Zhang, M. Mohammadi, R. Li, J. Shan, and S. L. Waslander. 2026. STaR: Scalable Task-Conditioned Retrieval for Long-Horizon Multi-Modal Robot Memory. *IEEE Robotics and Automation Letters*, 11(5), 5994–6001. <https://doi.org/10.1109/LRA.2026.3677723>
 - [193] C. Zhang, C. Yang, Q. Tan, J. Liu, A. Li, Y. Wang, J. Lu, J. Wang, and G. Yuan. 2025. Towards Memory-Efficient and Sustainable Machine Unlearning on Edge using Zeroth-Order Optimizer. *Proceedings of the Great Lakes Symposium on VLSI 2025*, 227–232. <https://doi.org/10.1145/3716368.3735273>
 - [194] C. Zhang, Z. Yang, J. Liu, Y. Li, Y. Han, X. Chen, Z. Huang, B. Fu, and G. Yu. 2023. AppAgent: Multimodal Agents as Smartphone Users. *arXiv preprint arXiv:2312.13771*. <https://arxiv.org/abs/2312.13771>
 - [195] H. Zhang, C. Xia, and Z. Wang. 2025. KVSwap: Disk-aware KV Cache Offloading for Long-Context On-device Inference. *arXiv preprint arXiv:2511.11907*. <https://arxiv.org/abs/2511.11907>
 - [196] Jiadong Zhang and Xiaosong Ma. 2026. MemArena: An Ego-Centric Benchmark for On-Device Agentic Personal Memory Assistants at Scale. *arXiv:2608.02613 [cs.CL]* <https://arxiv.org/abs/2608.02613>
 - [197] Jiaquan Zhang, Chaoning Zhang, Shuxu Chen, Zhenzhen Huang, Pengcheng Zheng, Zhicheng Wang, Ping Guo, Fan Mo, Sung-Ho Bae, Jie Zou, Jiwei Wei, and Yang Yang. 2026. Lightweight LLM Agent Memory with Small Language Models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 12914–12929. <https://doi.org/10.18653/v1/2026.acl-long.588>

- [198] Quqing Zhang, Kai Chen, Ning Liao, Zehao Lin, Bo Tang, Feiyu Xiong, Zhiyu Li, and Xiaoxing Wang. 2026. SparseX: Efficient Segment-Level KV Cache Sharing for Interleaved LLM Serving. arXiv:2606.01751 [cs.PF] <https://arxiv.org/abs/2606.01751>
- [199] T. Zhang, Y. Tong, J. Dong, K. Xu, Y. Wang, and J. Yuan. 2026. Forget by Uncertainty: Orthogonal Entropy Unlearning for Quantized Neural Networks. *Proceedings of the 43rd International Conference on Machine Learning*. <https://arxiv.org/abs/2602.00567>
- [200] Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2025. A Survey on the Memory Mechanism of Large Language Model-based Agents. *ACM Transactions on Information Systems* 43, 6, Article 155 (2025), 47 pages. <https://doi.org/10.1145/3748302>
- [201] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. ExpeL: LLM Agents Are Experiential Learners. *Proceedings of the AAAI Conference on Artificial Intelligence* 38, 17 (2024), 19632–19642. <https://doi.org/10.1609/aaai.v38i17.29936>
- [202] Xinkui Zhao, Qingyu Ma, Yifan Zhang, Hengxuan Lou, Sai Liu, Chang Liu, Guan jie Cheng, Naibo Wang, and Yueshen Xu. 2026. MUSE: A Heterogeneity-Aware Multimedia Search Engine for Mobile SoCs. arXiv:2511.19192 [cs.DC] <https://arxiv.org/abs/2511.19192>
- [203] Xiaochen Zhao, Kaikai Wang, Xiaowen Zhang, Chen Yao, and Aili Wang. 2026. HyMem: Hybrid Memory Architecture with Dynamic Retrieval Scheduling. arXiv:2602.13933 [cs.AI] <https://arxiv.org/abs/2602.13933>
- [204] Y. Zhao, B. Yuan, J. Huang, H. Yuan, Z. Yu, H. Xu, L. Hu, A. Shankarampeta, Z. Huang, W. Ni, Y. Tian, and J. Zhao. 2026. AMA-Bench: Evaluating Long-Horizon Memory for Agentic Applications. arXiv preprint arXiv:2602.22769. <https://arxiv.org/abs/2602.22769>
- [205] Y. Zhao, D. Zheng, K. Huang, Y. Wei, Z. Yang, and L. Zhou. 2026. MaskClaw: Edge-Side Personalized Privacy Arbitration for GUI Agents with Behavior-Driven Skill Evolution. arXiv preprint arXiv:2605.28646. <https://arxiv.org/abs/2605.28646>
- [206] J. Zheng, Z. Liu, Z. Shen, J. Qu, G. Chen, Y. Wang, Y. Xu, Y. Liu, and S. Cheng. 2026. Evaluating Memory Capability in Continuous Lifelog Scenario. In *Findings of the Association for Computational Linguistics: ACL 2026* (pp. 7063–7089). <https://doi.org/10.18653/v1/2026.findings-acl.351>
- [207] Y. Zheng, Z. Liu, X. Zhao, and J.-R. Wen. 2026. A Task-State Representation for Long-Horizon Mobile GUI Agents. arXiv preprint arXiv:2607.00502. <https://arxiv.org/abs/2607.00502>
- [208] H. Zhong, F. Faisal, L. França, T. Leesatapornwongsa, A. Szekeres, K. Rong, and S. Nath. 2026. ActionEngine: From Reactive to Programmatic GUI Agents via State Machine Memory. arXiv preprint arXiv:2602.20502. <https://arxiv.org/abs/2602.20502>
- [209] Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. MemoryBank: Enhancing Large Language Models with Long-Term Memory. *Proceedings of the AAAI Conference on Artificial Intelligence* 38, 17 (2024), 19724–19731. <https://doi.org/10.1609/aaai.v38i17.29946>
- [210] Z. Zhong, W. Yuan, L. Qu, T. Chen, H. Wang, X. Zhao, and H. Yin. 2026. Towards On-device Personalization: Cloud-device Collaborative Data Augmentation for Efficient On-device Language Model. *ACM Transactions on Intelligent Systems and Technology*, 17(1), 1-22. <https://doi.org/10.1145/3779452>
- [211] Sizhe Zhou and Jiawei Han. 2025. A Simple Yet Strong Baseline for Long-Term Conversational Memory of LLM Agents. arXiv:2511.17208 [cs.CL] <https://arxiv.org/abs/2511.17208>
- [212] Zhi Zhou, Xu Chen, En Li, Liekang Zeng, Ke Luo, and Junshan Zhang. 2019. Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing. *Proc. IEEE* 107, 8 (2019), 1738–1762. <https://doi.org/10.1109/JPROC.2019.2918951>
- [213] Lichen Zhu, Yueqian Lin, Yiheng Wang, Yudong Liu, Hai Li, and Yiran Chen. 2026. Will My Assistant Remember My Allergy? What Personal LLM Assistants Forget When Conversation Memory Is Compressed. arXiv preprint arXiv:2609.05767. arXiv:2609.05767 <https://arxiv.org/abs/2609.05767>
- [214] F. Ziliotto, J. R. Akkara, A. Daniele, L. Ballan, L. Serafini, and T. Campari. 2025. PersONAL: Towards a Comprehensive Benchmark for Personalized Embodied Agents. arXiv preprint arXiv:2509.19843. <https://arxiv.org/abs/2509.19843>
- [215] Wazeer Deen Zulfikar, Samantha W. T. Chan, and Pattie Maes. 2024. Memoro: Using Large Language Models to Realize a Concise Interface for Real-Time Memory Augmentation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Article 450, 18 pages. <https://doi.org/10.1145/3613904.3642450>