Review

Context Compression for LLM Agents: A Survey of Methods, Failure Modes, and Evaluation

Yifei Wang †, Ziteng Wang †, Yuling Shi, Silin Chen, Xinrui Wang, Yueqi Wang, Beijun Shen, Linjing Li, Xiaodong Gu, Julian McAuley, Daniel Dajun Zeng *

Posted Date: 29 May 2026

doi: 10.20944/preprints202605.2065.v1

Keywords: LLM agents; context compression; long-horizon tasks; agentic systems

 doi:10.20944/preprints202605.2065.v1

Review

Context Compression for LLM Agents: A Survey of Methods, Failure Modes, and Evaluation

Yifei Wang [1,2,†], Ziteng Wang [3,†], Yuling Shi [4], Silin Chen [4], Xinrui Wang [1,2], Yueqi Wang [5], Beijun Shen [4], Linjing Li [2], Xiaodong Gu [4], Julian McAuley [5] and Daniel Dajun Zeng [1,2,*]

[1] State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences
[3] Hefei University of Technology
[4] Shanghai Jiao Tong University
[5] University of California, San Diego
[*] Correspondence: dajun.zeng@ia.ac.cn
[†] These authors contributed equally.

Abstract

As Large Language Models (LLMs) evolve into autonomous agents for long-horizon tasks, managing unbounded interaction trajectories under fixed context budgets becomes a core systems challenge. Unlike standard long-context documents, agent trajectories are heterogeneous and interleave observations, reasoning traces, and tool executions, so compression must preserve temporal dependencies, actionable state, and structural fidelity. Yet existing methods remain fragmented, making it difficult to compare design choices and reason about their reliability implications. This survey introduces a unified taxonomy of agent context compression along three dimensions: *compression target* (what is compressed), *compression mechanism* (how it is transformed and retained), and *control policy* (who decides when compression is triggered). We further organize recurring failures in compressed execution into **F1**: *Pre-compression Decision Error*, **F2**: *In-compression Information Loss*, and **F3**: *Post-compression Access Failure*, and examine domain-specific trade-offs in software engineering, web navigation, and deep research. By unifying the design space, failure taxonomy, and evaluation perspective, this survey provides a foundation for building scalable and recoverable LLM agents.A collection of papers available at https://github.com/YerbaPage/Awesome-Context-Compression.

Keywords: LLM agents; context compression; long-horizon tasks; agentic systems

1. Introduction

With the rapid evolution of LLM agents, long context has emerged as a critical challenge [1,2] across open-ended domains like automated software engineering [3], visual GUI navigation [4,5], and deep research [6,7]. As agents continuously interact with dynamic environments, their operational history manifests as an **unbounded agentic trajectory**. This trajectory, typified by the **interleaved and heterogeneous** ReAct paradigm of Actions, Thoughts, and Observations (A-T-O) [8], rapidly exhausts the LLM's context window to trigger a severe ***context explosion***. Such an explosion precipitates a cascade of cognitive failures where information density plummets, critical constraints fade, and long-horizon planning collapses [9–11]. Blindly pursuing longer context windows does not eliminate the need for compression, as models are inherently inept at effectively utilizing vast amounts of uncompressed, noisy textual information [1,9], rendering **effective context management indispensable** for autonomy.

Context compression emerges as a highly effective pathway to mitigate this crisis, exemplified by real-world systems like ✳ Claude Code[1] that proactively compact contexts upon breaching predefined

[1] ✳ Claude Code: https://claude.com/product/claude-code.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 May 2026 doi:10.20944/preprints202605.2065.v1

trajectory thresholds. Unlike the compression of static long documents explored in prior literature [12–14], the profound **structural rigidity** of A-T-O sequences in agentic trajectories renders standard text reduction inapplicable. Consequently, as conceptualized in Figure 1, effective agent compression must **distill massive, noisy trajectories into compact, actionable states** while strictly **preserving causal action-observation dependencies** and structural integrity.

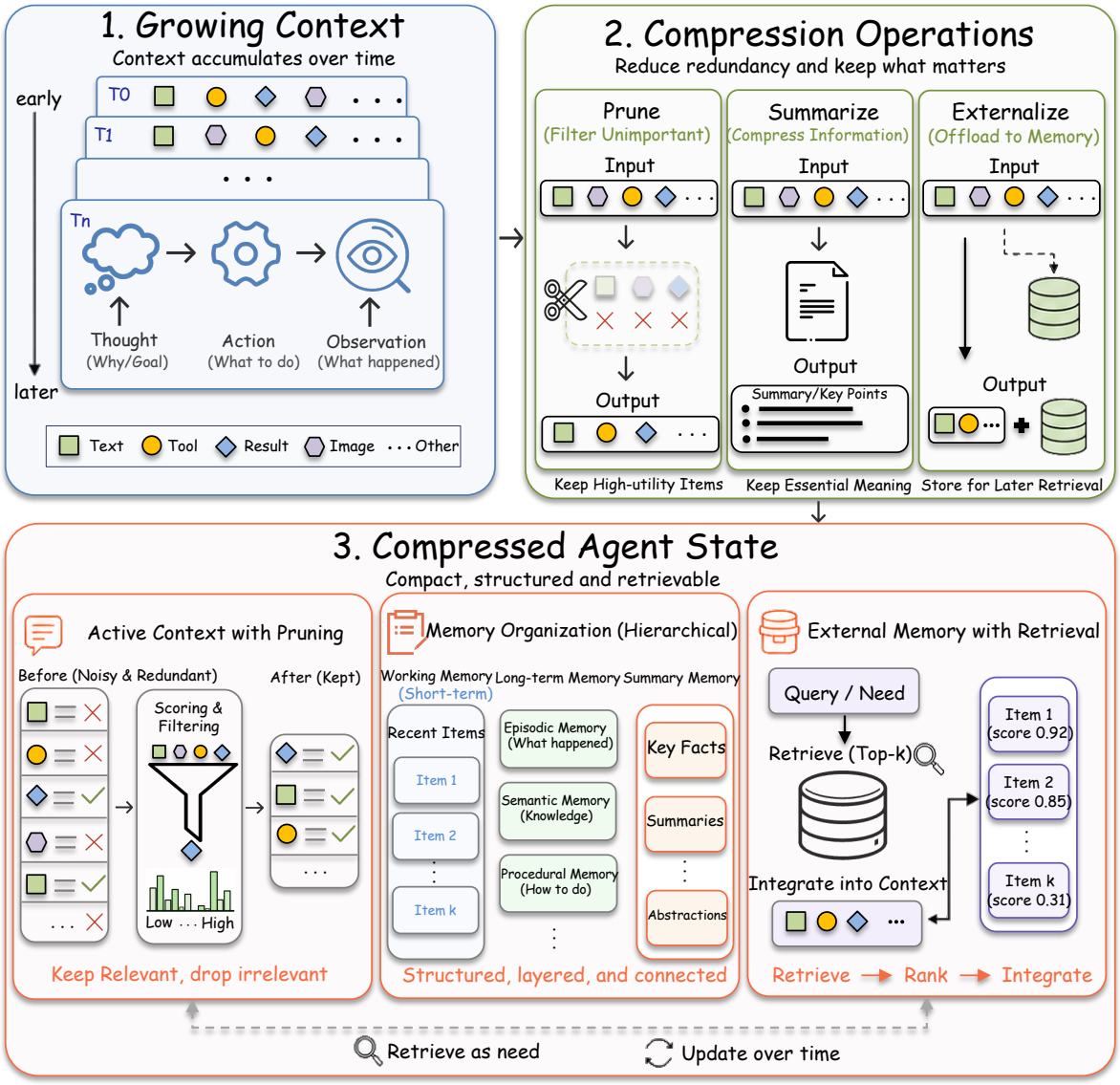

Figure 1. Illustration of the *context explosion* dilemma and the compression paradigm. Selective distillation transforms unbounded A-T-O sequences into structured representations while preserving operational fidelity.

The prevalence of autonomous agents has catalyzed a rapid proliferation of agentic context compression techniques. These methods are scattered across different compression targets, transformation mechanisms, and control strategies, ranging from pruning and summarization to selective retrieval and representation-level interventions [15–22]. Yet these efforts remain largely preliminary and disjointed explorations, lacking any **systematic synthesis** or **holistic perspective**. This conceptual fragmentation obscures a principled understanding of both the efficacy of underlying algorithmic designs and how specific compression choices trigger **distinct downstream agent failures** [23]. Consequently, practitioners are left without a theoretical framework to navigate the complex trade-offs among **context footprint**, **structural fidelity**, and **historical recoverability**.

Prior surveys on prompt compression [24,25] focus on the single-turn reduction of static inputs, whereas we study multi-turn trajectories whose relevance structure evolves over time. Work on KV

 doi:10.20944/preprints202605.2065.v1

cache [26–28] and model compression [29–31] addresses efficiency at the model or systems layer, while our concern is application-level context control during execution. Surveys on agent memory [32–34] organize methods by storage form, whereas we emphasize the intervention points, objects, and resulting forms of compressed context. Broader context engineering surveys span retrieval, memory, and orchestration at a high level [23]; by contrast, we provide a focused treatment of compression as a **first-class operational capability** in agent systems.

As agents enter the era of long-horizon tasks, context compression has become a three-dimensional systems problem rather than a single-turn prompt optimization trick. Yet a systematic review of this emerging domain remains absent. To fill this gap, we introduce a unified taxonomy organized around three questions: ***compression target*** (what is compressed), ***compression mechanism*** (how it is transformed and retained), and ***control policy*** (who decides when compression is triggered). These dimensions map directly onto the operational pipeline: targets specify **Select**, mechanisms instantiate **Compress** and **Store**, and policies govern when these stages, and when needed **Recover**, are activated. We further formalize three recurring failure modes, **F1**: *Pre-compression Decision Error*, **F2**: *In-compression Information Loss*, and **F3**: *Post-compression Access Failure*, and use them to expose reliability trade-offs. Finally, we critique current evaluation paradigms [35] and argue for metrics beyond end-task success, including information density, temporal consistency, and error propagation.

The rest of the survey is organized as follows. Section 2 introduces the background and unified pipeline, Section 3 presents a formal lens, Sections 4 and 5 develop the taxonomy and failure analysis, and the remaining sections discuss domain-specific trade-offs, evaluation, future directions, and conclusion.

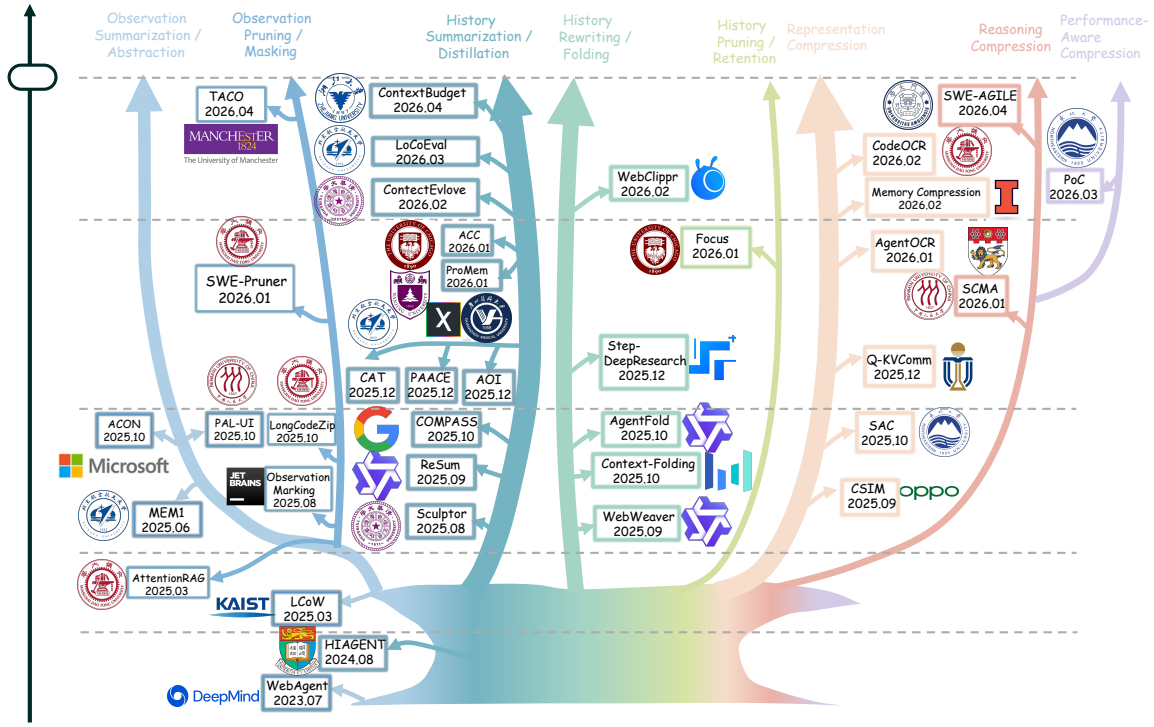

Figure 2. A design-space view of agent context compression. The figure synthesizes targets, mechanisms, and control policies into a method-level landscape, grouping approaches by compression semantics while preserving chronological evolution. See Appendix A and B for methodology, taxonomy details, and open-source resources.

2. Redefining Context: From Static Prompts to Dynamic Trajectories

2.1. What Is Unique About Agent Context

Formally, agent context is a time-indexed state C_t continuously updated by interaction tuples $x_t = (a_t, t_t, o_t)$ of actions, thoughts, and observations [8]. Unlike standard long-text processing, this dynamic trajectory exhibits four fundamental properties that distinguish it from static prompts:

 doi:10.20944/preprints202605.2065.v1

Property 1: Dynamic Growth. Context is not provided all at once but expands incrementally by absorbing new interactions x_t with each agent step. Unlike static documents with fully exposed structures, agentic compression operates blindly on this continuous stream, dynamically distilling history without foreseeing future states.

Property 2: Heterogeneous Composition. Context comprises highly distinct structured content, including natural language plans, code snippets, JSON tool outputs, screenshots, and error logs. While natural language thoughts (t_t) may endure lossy abstraction, actions and observations often demand strict syntactic fidelity, mandating type-aware rather than uniform compression.

Property 3: Multi-step Dependency. Information compressed or discarded at a certain step may only be needed several steps later. Driven by continuous streaming, the utility of any specific tuple component e manifests over future steps: $\mathcal{U}(e) = \mathbb{E}\left[\sum_{\tau=t}^{T} \gamma^{\tau-t} \cdot u_\tau(e)\right]$. Consequently, it is inherently difficult to predict future demands during compression, as the true importance of e remains unobservable at time t.

Property 4: Error Propagation. Errors or omissions introduced by compression are not discovered locally. Instead, a state distortion $\delta_t = d(C_t, \tilde{C}_t)$ propagates backward along the trajectory: $C_{t+k} = U^{(k)}(\tilde{C}_t, x_{t+1:t+k})$. These errors accumulate irreversibly and are treated as factual grounding by subsequent reasoning steps, permanently shifting the action space and derailing the agent from the optimal path.

► ***Why Traditional Compression Fails*** ◄ Traditional methods satisfy one-off budget constraints by optimizing for immediate relevance [24,36]. For example, KV cache compression typically evicts tokens with low current attention scores [14,37]. However, under the four properties above, applying such static interventions to an evolving agentic loop triggers compounding distortions, as continuous interaction defies any static definition of information importance.

2.2. A Unified Operational Pipeline for Agent Context Compression

We formulate a unified four-stage pipeline that decomposes compression into an operator system over the context lifecycle:

$$C_{t+1} = \underbrace{R(q_{t+1}, \mathcal{M}_t)}_{\text{Recover}} \oplus \underbrace{M\Big(\Phi\big(S(C_t)\big)\Big)}_{\text{Store}\leftarrow\text{Compress}\leftarrow\text{Select}} \oplus x_{t+1} \tag{1}$$

- **Select (S):** A critical triage phase where the system determines which subsets of the context require compression and which demand exact preservation.
- **Compress (Φ):** Specific reductive or structural transformations, ranging from simple truncation to latent encoding, are executed on the selected subset.
- **Store (M):** The resulting compressed representations are committed either to the active working memory or offloaded to an external storage state $\mathcal{M}_t$.
- **Recover (R):** Driven by the necessity of deferred utility, relevant compressed information is selectively retrieved and decoded from $\mathcal{M}_t$ using the current query q_{t+1}.

3. Compression as Sufficient State Approximation

We argue that agent context compression is best understood as a sufficient-state approximation problem: the goal is to maintain the shortest representation that remains sufficient for future action.

Compression as State Transformation.

Let C_t denote the current context state. We treat each compression method as a transformation

$$O_i : \mathcal{C} \to \mathcal{C}, \qquad C'_t = O_i(C_t),$$

where O_i maps C_t to a shorter, reorganized, or externally mediated representation C'_t under task-specific constraints. This abstraction is deliberately broad. It covers the operator families discussed later

 doi:10.20944/preprints202605.2065.v1

in the survey, including truncation, pruning, summarization, externalization, and representation-level compression.

Ideal Sufficient Memory.

From an algorithmic-information perspective, these operators can be viewed as practical approximations to an ideal sufficient memory. Let E_t denote the currently accessible environment state, A the agent policy or executor, τ the task or task distribution, and ϵ the maximum acceptable degradation in downstream performance. We define the ideal sufficient memory length as

$$L_\epsilon(C_t) = \min_{\tilde{C}_t \in \mathcal{S}_t} K(\tilde{C}_t \mid E_t) \quad \text{s.t.} \quad \Delta_\tau(A; C_t, \tilde{C}_t, E_t) \leq \epsilon, \tag{2}$$

where $\mathcal{S}_t = \mathcal{S}(C_t, E_t)$ denotes the admissible space of compressed states derived from C_t under environment E_t. Here $K(\tilde{C}_t \mid E_t)$ is standard conditional Kolmogorov complexity, and $\Delta_\tau(A; C_t, \tilde{C}_t, E_t)$ measures the loss in future task performance induced by replacing C_t with $\tilde{C}_t$. The quantity L_ϵ is not meant as a computable training objective; rather, it serves as a stylized lower bound. A compressed state is useful only insofar as it remains compact, task-sufficient, and recoverable for future decisions.

From Approximation to Failure.

This objective turns the next two sections into one question: how does a method approximate a sufficient future state, and where can that approximation fail? Truncation bets that dropped spans have no future value. Pruning preserves an exact sufficient subset. Summarization replaces evidence with an executable abstraction. Externalization keeps part of the sufficient state off-context and shifts the bottleneck to recovery. Representation-level compression pushes sufficiency into latent space and sacrifices transparency. Under this view, the taxonomy compares approximation strategies, while the failure modes identify where sufficiency or recoverability is lost.

4. A Taxonomy of Agent Compression

4.1. From Pipeline to Design Space

The unified pipeline in § 2.2 can be read as a comparative design space for agent compression. We organize the literature along three axes: *what* is selected for compression, *how* it is transformed, and *who* decides when compression occurs. These axes map naturally onto the pipeline: targets define the input to **Select** (S), mechanisms implement **Compress** (Φ) and **Store** (M), and control policies govern when these stages, and when necessary **Recover** (R), are invoked.

4.2. Compression Targets (What)

We begin with the objects processed by the **Select** (S) operator. Table 1 summarizes these targets in terms of granularity, information density, causal coupling, and recoverability. Together, these properties determine both compressibility and likely failure symptoms.

 doi:10.20944/preprints202605.2065.v1

Table 1. A comparative anatomy of compression targets. Each target is characterized by its granularity, information density, causal coupling, and recoverability, which collectively shape its compressibility and its failure symptom. Here, ↑ and ↓ indicate a stronger or weaker degree respectively, and repeated arrows indicate a higher degree of that tendency.

Target	Granularity	Information Density	Causal Coupling	Recoverability	Typical Failure Symptom
Observation	Token / Segment	★★	★★	★★★	Structural Distortion
Trajectory	Episode	★	★	★★	Planning Drift
Plan & Reasoning	State	★★★	★★★★	★	Metacognitive Error
Memory State	State	★★★	★★★	★	Retrieval Blindness
Representation	Token / Latent	★★★★	—	—	Semantic Collapse

4.2.1. Observation Compression

Observation compression targets raw inputs, such as tool outputs, HTML DOMs, logs, or screenshots. These inputs are high-volume and redundant, but a single omitted selector, trace line, or visual cue may determine the next action. Methods such as The Complexity Trap [15] therefore treat it as a low-cost front-end intervention, typically via masking or shallow filtering.

4.2.2. Trajectory Compression

Trajectory compression focuses on accumulated multi-step histories, including failed attempts, redundant tool calls, and repetitive loops. Its main difficulty is deferred utility: information that appears irrelevant at step t may become indispensable later. As a result, trajectory compression is especially vulnerable to **F2**: *In-compression Information Loss* and delayed downstream inconsistency. Representative approaches include controller-mediated reduction in AgentDiet [16], periodic summarization in ReSum [17], and proactive folding in AgentFold [18].

4.2.3. Plan and Reasoning Compression

Plan and reasoning compression targets internal artifacts, including subgoals, deliberative traces, self-reflections, and task decompositions. These representations are compact but brittle: dropping a seemingly minor premise can invalidate an entire downstream plan. This target is therefore sensitive to **F1**: *Pre-compression Decision Error*, because the system must correctly judge which reasoning artifacts remain decision-critical. Representative methods include feedback-driven summarization in ACON [38] and agent-initiated memory management in Focus [39].

4.2.4. Memory State Compression

Memory state compression maps an unbounded interaction stream into a fixed-capacity representation that survives beyond the immediate context window. It most explicitly realizes the **Store** (M) stage, so its main bottleneck is accessibility rather than faithfulness. Once information is offloaded, the system must still recognize when that memory matters and reconstruct it at use time. Consequently, memory-state compression is most directly exposed to **F3**: *Post-compression Access Failure*. Learned memory policies such as MEM1 [40] address this challenge by coupling state formation with future retrieval utility.

4.2.5. Representation-Level Compression

Representation-level compression operates below the natural-language interface, targeting KV caches, multimodal token embeddings, or other latent states. It often delivers the strongest efficiency gains, but with minimal recoverability and weak inspectability. These methods therefore shift

 doi:10.20944/preprints202605.2065.v1

the problem from semantic editing to opaque state management. Representative examples include AgentOCR [21], Q-KVComm [41], and SAC [42].

4.3. Compression Mechanisms (How)

Having identified the targets, we now turn to the transformations themselves. What most distinguishes compression mechanisms is how they relate the compressed result to the original context. Some remove content, some paraphrase it, some retain an exact subset, and others store information so that it can be recovered later. This distinction is crucial because methods with similar compression ratios may differ sharply in *faithfulness*, *inspectability*, and *downstream recoverability*.

4.3.1. Masking and Truncation

These operators keep a prefix or suffix of the context, or selectively mask designated spans:

$$O_{\text{trunc}}(\mathcal{C}) = \mathcal{C}[:k],\ O_{\text{mask}}(\mathcal{C}) = \text{mask}(\mathcal{C}, S).$$

These operators, which directly modify the support of C through span removal or placeholder substitution, are universal, low-overhead, and easy to deploy. That same directness also defines their limitation: once content is removed, recoverability is lost unless another subsystem stores it elsewhere. Such operators are therefore best suited to settings in which marginal utility decays rapidly with recency or position, as in sliding-window baselines or hard budget enforcement [43].

4.3.2. Summarization and Abstraction

These operators rewrite a long context into a shorter semantic representation:

$$O_{\text{sum}}(\mathcal{C}) = \hat{\mathcal{C}},\ |\hat{\mathcal{C}}| \ll |\mathcal{C}|.$$

Unlike truncation, summarization preserves utility by semantic rewriting rather than direct deletion. This often yields substantially higher information density and better long-horizon compactness, which explains its popularity in research and search agents. The trade-off is that faithfulness becomes model-dependent: once context is rewritten, omissions, over-generalizations, and spurious inferences may be introduced, making summarization a primary locus of **F2**: *In-compression Information Loss*. Representative systems include LCoW [44], COMPASS [45], and ReSum [17].

4.3.3. Pruning and Reduction

Pruning removes content selectively while preserving the remaining structure exactly:

$$O_{\text{prune}}(\mathcal{C}) = \mathcal{C} \setminus R,$$

where R is a subset selected by a relevance, dependency, or structural criterion. Unlike summarization, pruning leaves retained evidence untouched. It is therefore especially valuable when syntactic form, pointer alignment, or execution traces must remain exact, as in coding and GUI settings. In exchange, pruning typically compresses less aggressively than summarization unless the context contains substantial redundancy. Representative examples include SWE-Pruner [46] and WebClipper [47].

4.3.4. Externalization and Retrieval

This mechanism splits context into a working state and an external store:

$$O_{\text{ext}}(\mathcal{C}) = (\mathcal{C}', M),\ O_{\text{ret}}(\mathcal{C}, M) = \mathcal{C} \oplus r(M).$$

It is the clearest realization of **Store** (M) and **Recover** (R) in the pipeline. By turning part of compression into retrieval, it permits aggressive footprint reduction so long as critical information can later be reconstructed on demand. It is therefore the only mechanism that combines high compression with principled recoverability. Its main risk shifts toward **F3**: *Post-compression Access Failure*: not corruption, but missing, mistimed, or incorrect re-access. Systems such as Step-DeepResearch [48], CAT [49], and WebWeaver [50] rely heavily on this paradigm.

 doi:10.20944/preprints202605.2065.v1

4.3.5. Representation Compression

Operating on latent structures, $O_{\text{repr}}(\mathcal{C}) = \tilde{C}$ compresses history without explicit textual rewriting. These methods optimize efficiency most directly, but they do so by moving compression beneath the symbolic interface available to the agent and the researcher. As a result, they offer limited interpretability, weak controllability, and little explicit support for exact recovery. Methods such as DebateOCR [22] illustrate this trade-off.

4.4. Control Policies and Intervention Timing (Who and When)

Control policy, not compression mechanism, determines when intervention occurs. The same operator may be triggered reactively at a budget threshold, periodically by an external controller, or proactively by the agent itself. We therefore separate operator family from control policy: policy specifies when **Select**, **Compress**, and, where applicable, **Recover** are invoked, who makes that decision, and which signals are sufficient to trigger action.

4.4.1. System-Controlled Policies

System-controlled policies apply fixed rules to trigger compression:

$$\pi_{\text{sys}} : \mathcal{C} \mapsto (i, \theta).$$

Depending only on observable signals such as length, position, or step count, they are predictable, cheap, and easy to benchmark. Their weakness is semantic blindness: they intervene because a budget is reached, not because task utility has changed. Such policies therefore dominate early and production-constrained systems, including predefined schedules in AgentDiet [16] and threshold-based compaction regimes.

4.4.2. External Controller Policies

External-controller policies delegate the decision to a separate module:

$$\pi_{\text{ext}} : \mathcal{C} \to (i, \theta).$$

The controller may be a smaller model, a planner, or a retrieval manager that observes the evolving context and selects a compression action for the main agent. This decoupling improves modularity and sometimes yields more stable control, because compression can be optimized independently of task execution. The cost is additional latency, token overhead, and a new interface on which information can be lost. Examples include PAL-UI [51] and AOI [52].

4.4.3. Agent-Controlled Policies

Agent-controlled policies make compression part of the agent's own action space:

$$\pi_{\text{agent}}(C_t, s_t) \to (i, \theta),$$

where s_t denotes the agent's internal state. This policy class is the most semantically aware, because the same system that reasons about the task also decides which context should be compressed, preserved, or revisited. It is therefore well suited to proactive memory management and self-reflective intervention, as in CAT [49], Focus [39], and ContextBudget [53]. Its weakness is precisely the agent's own fallibility: errors in self-assessment readily manifest as **F1**: *Pre-compression Decision Error*.

4.4.4. Learned Compression Policies

Learned policies optimize compression behavior from data:

$$\pi_{\text{learn}} = \arg\max_{\pi} \mathbb{E}_{\tau \sim \pi}[R_{\text{task}}(\tau) - \lambda \, \text{Cost}(\tau)].$$

Whether trained by supervised fine-tuning or reinforcement learning, they seek to align compression decisions with downstream task return rather than handcrafted heuristics. This makes them the most expressive policy class and, in principle, the closest approximation to the ideal sufficient-memory objective in Section 3. In practice, however, they are also the most resource-intensive and the most

 doi:10.20944/preprints202605.2065.v1

sensitive to reward misspecification. Representative examples include ACON [38], MEM1 [40], and related policy-learning approaches.

5. Failure Modes

We organize context compression failures by the earliest stage at which they arise: **F1**: *Pre-compression Decision Error*, **F2**: *In-compression Information Loss*, and **F3**: *Post-compression Access Failure*, as shown in Figure 3.

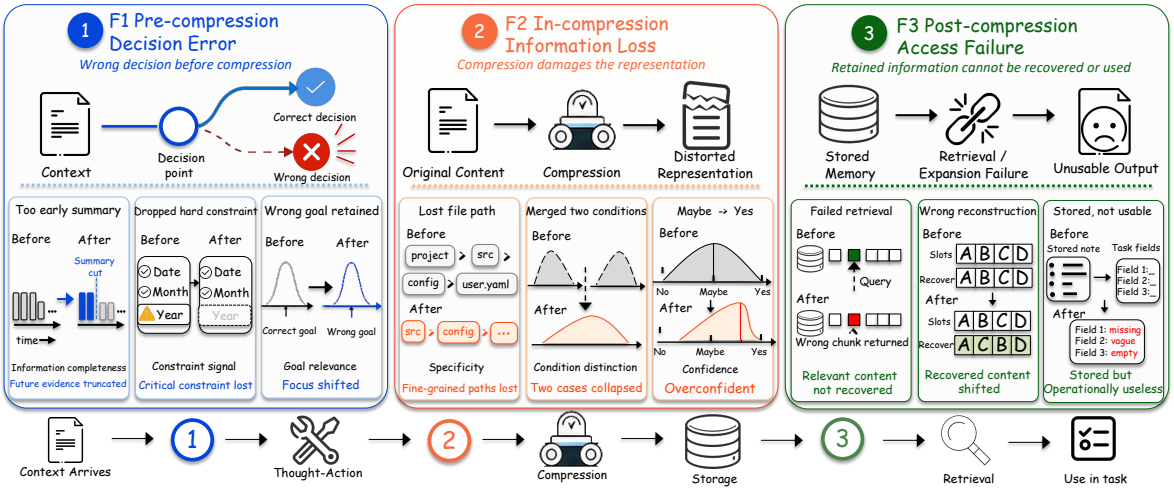

Figure 3. A temporal taxonomy of context compression failures. Failures are classified by the earliest point at which they occur in the compression pipeline.

 doi:10.20944/preprints202605.2065.v1

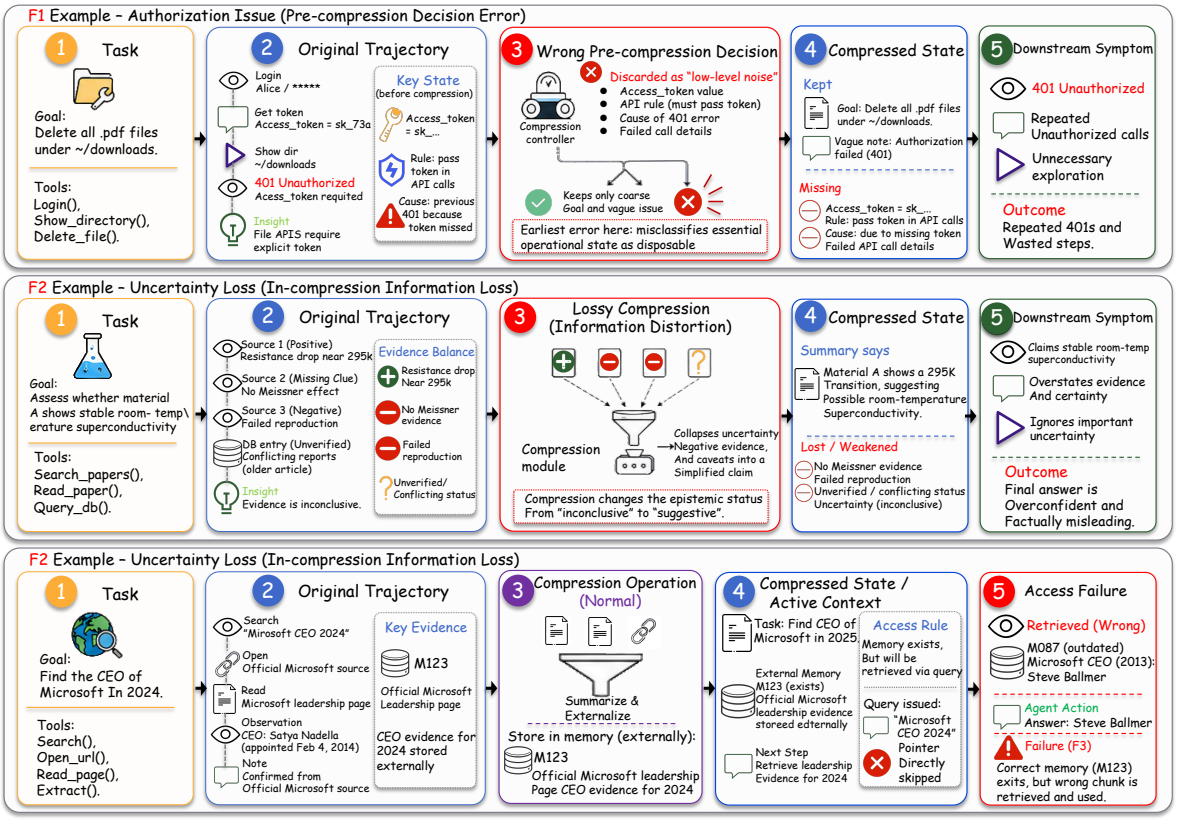

Figure 4. Three illustrative cases of context compression failure modes. F1 shows a pre-compression decision error, where the compression controller misclassifies authentication state and API-call details as low-level noise before summarization. F2 shows in-compression information loss, where a compression module distorts the evidential balance by weakening uncertainty, negative evidence, and caveats. F3 shows a post-compression access failure, where the correct memory item exists after compression but is not correctly expanded or retrieved, causing the agent to use an outdated wrong chunk.

F1: ***Pre-compression Decision Error*** refers to failures that arise before compression itself, when the system chooses the wrong moment, target, or granularity of compression. The issue is not the compression operator, but a pre-compression decision that already departs from task requirements. Typical cases include premature compression [49,54], misjudging the retention value of critical information [21,55], or compressing constraints that should remain exact [49]. A representative case, detailed in Appendix D, removes authentication-related state before the agent reaches the step where it becomes necessary.

F2: ***In-compression Information Loss*** refers to failures introduced by the compression operation itself, where representation transformation corrupts semantics, structure, relations, constraints, or fine-grained detail. The central question is whether compression preserves the original information faithfully enough for later use. Typical cases include omitted semantics [54,56], disrupted relational structure [19], oversimplified constraints [49,56], or compressed uncertainty that becomes an unwarranted conclusion [54]. Appendix D includes a trajectory-level case in which compression preserves the broad task intent but loses a fine-grained constraint, leaving the later agent state semantically incomplete.

F3: ***Post-compression Access Failure*** refers to failures of post-compression accessibility: information is retained in some form, but cannot be correctly retrieved, expanded, or reconstructed when needed. The issue is therefore recoverability rather than pre-compression choice or in-compression corruption. Typical cases include failed retrieval [56–58], incorrect reconstruction [56], or compressed states that remain storable yet functionally unusable [57,58]. The Appendix D case makes the same point from the access side: the correct compressed memory exists, but later retrieval returns a semantically similar yet wrong memory item.

 doi:10.20944/preprints202605.2065.v1

Together, **F1–F3** define a temporal failure taxonomy over the compression pipeline. For hybrid cases, we assign the label to the earliest failure that causally triggers later errors, since agent workflows naturally exhibit error propagation [59].

6. Domain-Specific Analysis

Compression is domain-dependent: coding agents require structural fidelity and state continuity; web and GUI agents must preserve heterogeneous multimodal observations; DeepResearch agents depend on deferred evidence recovery across long horizons. Figure 5 summarizes these regimes.

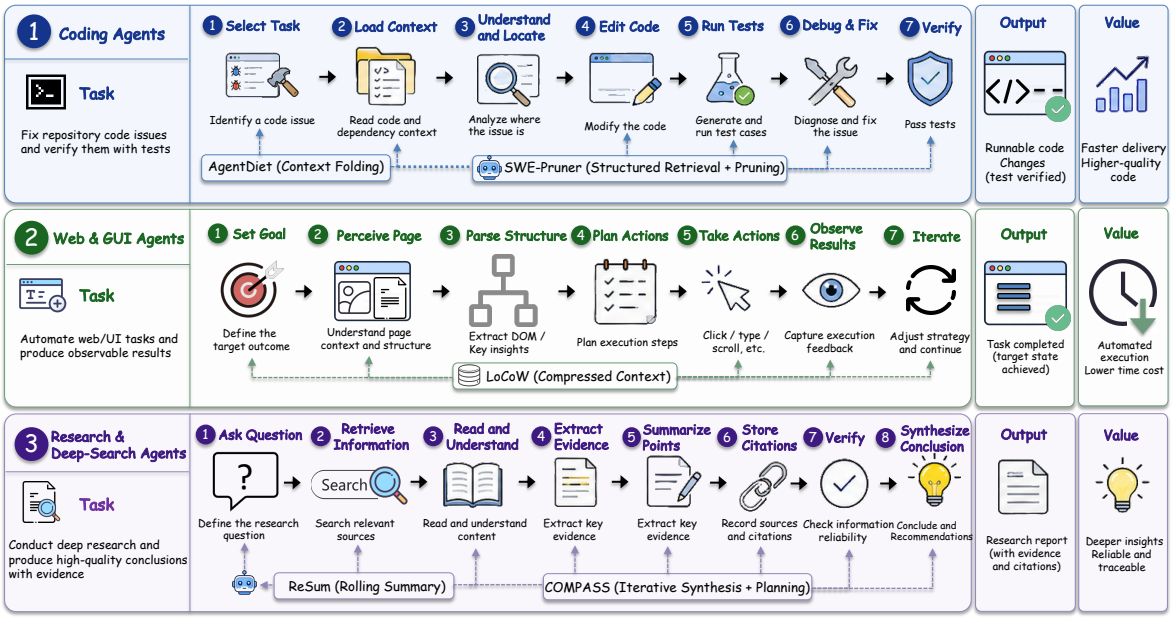

Figure 5. Domain-specific context compression regimes, contrasting setup, workflow, interaction loop, dominant risks, and strategy choices.

Coding Agents Coding places the strongest pressure on structure preservation, exact evidence retention, and reliable recovery. Accordingly, methods such as SWE-Pruner [46], LongCodeZip [60], and CAT [49] emphasize structure-aware pruning and explicit state retention. Table 2 compares recent production-grade coding agents through the unified operational pipeline, using only public evidence and high-confidence product signals. The contrast is instructive. Claude Code emphasizes **Compress** through reactive compaction, Cursor places more weight on **Store** and **Recover** through repository indexing and retrieval, and Codex appears most grounded in **Select** plus workspace continuation. Across all three, the common pattern is not aggressive lossy shortening, but coordinated pipeline control under tight context budgets.

Table 2. Pipeline-level comparison of recent production-grade coding agents, organized by the stages in Section 2.2.

System	Select (S)	Compress (Φ)	Store (M)	Recover (R)
Claude Code	Reactive, threshold-based selection.	Auto-compaction of prior interaction history.	Compacted state retained in active context.	Implicit recovery via continued execution.
Cursor	Retrieval-first selection of files, code blocks, and rules.	Relevance filtering and contextual assembly.	Repository index, project rules, and workspace context.	Re-injection of retrieved files, rules, and evidence.
OpenAI Codex	Task-grounded selection over files, instructions, and threads.	Session continuation and workspace grounding.	Task threads, instruction files, and repository state.	Recovery through continued threads and file re-access.

Web & GUI Agents Web and GUI agents operate over heterogeneous, partially redundant observations. Methods such as LCoW [44] and PAL-UI [51] exploit redundancy, but representation-level compression must still preserve modality and interface cues.

 doi:10.20944/preprints202605.2065.v1

DeepResearch Agents Research and deep-search agents are defined less by structural fidelity than by delayed evidence use. Their main risk is that abstraction hides evidence that matters only later. ReSum [17] and COMPASS [45] favor summarization, whereas Step-DeepResearch [48] and WebWeaver [50] externalize or retrieve evidence to preserve long-horizon dependencies.

7. Beyond Task Success: Rethinking Evaluation

Outcome-Centric Evaluation. Benchmarks still judge context compression mainly by downstream task success. Success rate, with token count or compression ratio as an auxiliary proxy, remains the default protocol. This is convenient but inadequate: success entangles compression with planning and execution, while shorter prompts reveal little about whether task-critical evidence survives. Cross-paper comparison is further blurred by changes in backbone, task set, and context budget.

The Missing Object. The missing object is not the task outcome but the compressed state itself. For agents, four dimensions are decisive: *information density*, *recoverability*, *error propagation*, and *controller overhead*. Together, they ask whether compression preserves task-critical evidence, supports later restoration, avoids long-horizon failure amplification, and remains efficient enough to justify its control cost.

From Outcomes to State Quality. We therefore evaluate compressed states through $Q = (D, R, P, O)$, where D denotes task-relevant density, R recoverability, P propagation risk, and O operational overhead, and summarize comparison through $J = \alpha D + \beta R - \gamma P - \delta O$. The point is simple but consequential: compression should be judged as state quality, not only task outcome. This reframing ties benchmark outcomes to compression quality under domain-specific constraints Constraint_i. Taken together, this reframing motivates a broader discussion of recoverable, faithful, and long-horizon context management, which we defer to Appendix E.

8. Conclusion

Agent context compression is no longer just a preprocessing convenience. For long-horizon agents, it is a state-management problem over a changing workspace, not a one-shot prompt-shortening step. This survey structures the area along three axes: targets, mechanisms, and control policies. It also introduces a failure taxonomy and an evaluation framework that goes beyond task success. The key challenge is not only stronger compression, but compression that remains recoverable, faithful, and aligned with long-term agent execution.

Limitations

Coverage.

Because agent context compression is evolving rapidly, our synthesis is intended to be representative rather than exhaustive, especially for the newest systems and engineering practices still stabilizing in the literature.

Scope.

We deliberately focus on context compression during agent execution, rather than adjacent areas such as general long-context modeling, context engineering, memory management, or model-level KV-cache optimization, except where they directly inform agent compression. We view these neighboring directions as natural extensions and leave a unified treatment to future work.

Abstraction Level.

Our taxonomy prioritizes conceptual unification across targets, mechanisms, policies, failure modes, and evaluation. This improves comparability, but necessarily abstracts away some implementation-level variation across backbone models, task settings, and reporting protocols.

 doi:10.20944/preprints202605.2065.v1

We view these limitations primarily as boundaries of scope rather than weaknesses of the framework, and hope the structure offered here provides a stable basis for more exhaustive updates as the field matures.

AI Usage Disclosure: The authors used AI-based writing assistants to improve grammar, phrasing, and local clarity during manuscript preparation. All AI-assisted suggestions were reviewed, revised, and selectively incorporated by the authors, who take full responsibility for the final text, technical framing, citations, and conclusions.

Ethical Statement: This work is a survey of publicly available research papers, technical reports, product documentation, and open-source resources on context management for LLM agents. It does not involve private data, human subjects, or sensitive personal information. We aim to represent prior work accurately and cite sources faithfully; any discussion of deployed systems is based only on publicly accessible evidence and is used solely for scholarly analysis.

Appendix A. Methodology of the Survey

Appendix A.1. Search Strategy

Because agent context compression is an emerging topic without a unified terminology, we adopted a broad search strategy rather than relying on a single keyword. We searched ACL Anthology, DBLP, arXiv, Google Scholar, Semantic Scholar, and major NLP, machine learning, software engineering, and agent-system venues. The search covered both explicit compression terms and adjacent terms used in related communities.

Specifically, we used keyword combinations such as *"LLM agent"* with *"context compression"*, *"trajectory compression"*, *"history summarization"*, *"context pruning"*, *"context folding"*, *"agent memory compression"*, *"memory retrieval"*, and *"long-horizon agent context"*. We also searched domain-specific terms, including *"coding agent context"*, *"web agent memory"*, *"GUI agent observation compression"*, and *"deep research agent memory"*. These queries were designed to capture papers that may not use the exact phrase *context compression* but nevertheless reduce, rewrite, externalize, or retrieve agent history under a limited context budget.

In addition to keyword search, we performed backward and forward citation tracing from representative work on prompt compression, long-context modeling, agent memory, web agents, coding agents, and deep-research agents. This step was necessary because many relevant systems describe compression-related operations as *memory management*, *state retention*, *context engineering*, *trajectory summarization*, or *retrieval-based continuation*. We therefore treated terminology flexibly and focused on whether a method changes what historical information remains available to the agent during future execution.

Appendix A.2. Inclusion and Exclusion Criteria

We included a paper or system if it satisfied at least one of the following conditions. First, it proposes a method that compresses, filters, summarizes, prunes, folds, externalizes, or retrieves the context of an LLM agent during multi-step execution. Second, it introduces a memory or retrieval mechanism whose purpose is to preserve task-relevant information under a finite context budget. Third, it analyzes long-horizon agent behavior in a way that reveals failures caused by context reduction, memory selection, retrieval, or compressed-state reuse. Fourth, it describes a deployed or production-grade agent system with publicly documented context-management behavior relevant to the operational pipeline studied in this survey.

We excluded work whose primary contribution lies outside application-level agent context management. In particular, we excluded methods that only compress static, single-turn prompts unless they are explicitly adapted to multi-step agent trajectories. We also excluded model-level compression methods, such as weight quantization, model pruning, distillation, and generic KV-cache optimization,

 doi:10.20944/preprints202605.2065.v1

when they do not alter the information available to the agent as part of its executable context. Similarly, general retrieval-augmented generation was not treated as agent context compression unless retrieval is coupled to an evolving agent state, trajectory history, or memory store.

For borderline cases, we used a functional criterion: a work was considered relevant if its method changes how past observations, actions, thoughts, plans, or memory states are preserved and reused for future decisions under a context constraint. This criterion allowed us to include systems that do not explicitly use the term *compression* but still perform compression-like state management. When a method combined multiple components, such as summarization plus retrieval or pruning plus external memory, we categorized it according to the component that most directly determines the compressed state available to later agent execution.

Finally, for production-grade systems and technical reports, we only relied on public evidence and high-confidence descriptions. When implementation details were unavailable, we characterized the system at the level of observable pipeline behavior rather than making assumptions about proprietary mechanisms.

Appendix B. Extended Taxonomies and Resources

Appendix B.1. Details About Taxonomy

In the main text, we characterize context management methodologies based on three core dimensions, specifically *when*, *what*, and *how*, while further analyzing their application *domain* as a complementary perspective. Table A3 below instantiates this framework at the level of individual methods. Specifically, the *when* column indicates the trigger mechanism and decision maker for context compression or management, including system-controlled, external-controller, agent-controlled, and learned policies; the *what* column identifies the context object being managed, such as observations, trajectories, memory states, plans and reasoning traces, or representation-level signals; and the *how* column summarizes the compression mechanism, including truncation and masking, pruning and reduction, summarization and abstraction, externalization and retrieval, and representation-level compression. In addition to these three dimensions, we report the *domain* of each method to indicate its primary task setting, and we analyze its distribution and characteristics separately in the main text. Taken together, the table provides a fine-grained view of how different methods are positioned within the taxonomy and how their design choices vary across task domains.

Appendix B.2. Open-Source Standouts

To more comprehensively reflect the openness and reproducibility of research in this area, we further collected and summarized the open-source methods included in this survey, as shown in Table A4. The table exclusively incorporates methods accompanied by publicly available code, presenting them across three columns: method name, open-source link, and publication year. This organization enables readers to efficiently locate the corresponding implementations and repository resources. As shown in the table, the current open-source work is concentrated in a small number of representative methods, and the release years span 2025 to 2026, which indicates that this field is still evolving rapidly and that its open-source ecosystem is steadily maturing. Overall, this statistics not only improves the reproducibility of our survey results, but also provides a direct entry point for future researchers to conduct method comparison, code replication, and further development.

Appendix B.3. Datasets & Benchmarks

Table A5 summarizes the representative datasets and benchmarks collected in `datasets_info.md`. We organize them by task family and retain only the information most useful for a survey on agentic context management: benchmark purpose, publication venue/year, and public availability. To keep the table compact, we move the task-focus and scale description into the benchmark cell itself, so each row reads as a short self-contained summary.

 doi:10.20944/preprints202605.2065.v1

The *Benchmark* cell includes the dataset name, a citation to the original paper, and a brief description of the task and scale. The *Source and year* column records the original publication venue or release year, while the *Open Source* column provides the public repository or dataset URL when available, and indicates the absence of public releases otherwise. Benchmarks with closely related variants are grouped together to keep the presentation compact.

First, the selected benchmarks are intentionally diverse, encompassing software engineering, web browsing, office automation, mobile apps, deep research, long-context reasoning, memory, code generation, and long-document summarization. This diversity is essential, as context management methods frequently navigate the trade-off between retaining essential task-critical evidence and pruning redundant historical data, with individual benchmarks typically evaluating only a specific subset of these behaviors.

Second, numerous benchmarks are structured as closely related families or variants. For instance, *SWE-bench Verified* [61] and *SWE-bench Lite* [61] assess software issue resolution under different difficulty and edit constraints; *BrowseComp* [62], *BrowseComp-Plus* [63], and *BrowseComp-ZH* [64] cover browsing-centric deep search in different settings; *Mind2Web* [65] and *Multimodal-Mind2Web* [66] differ by adding visual grounding; *LongBench* [2], *LongBench V2* [67], *BABILong* [68], *NeedleBench* [69], and *NIAH* [70] probe long-context handling from complementary angles.

Third, in instances where a benchmark entry represents a comprehensive framework or family rather than an isolated dataset, we designate it by its representative name in the table and elucidate the included variants within the accompanying notes. This is especially helpful for benchmarks such as *MRQA* [71], *Loghub* [72], and *CompileBench / CRUST-Bench* [73], where a single umbrella name aggregates multiple sub-datasets or tasks.

Finally, the *Open Source* column is interpreted broadly, providing the public dataset or repository URL when available, and explicitly noting cases where no public release could be identified within the collected sources. The original publication year and venue are preserved in the *Source and year* column.

Appendix C. Implementation Details of Compression Operators

This appendix supplements the taxonomy in Section 4.3 by summarizing concrete implementation details of representative compression operators. We focus on three methods with relatively explicit descriptions: ReSum [17] and AgentFold [18] for summarization and abstraction, and SWE-Pruner [46] for pruning and reduction. The goal is not to reproduce full prompts verbatim, but to identify the prompt roles, response templates, algorithmic steps, and compression-specific design choices that instantiate the operator families discussed in the main text.

Appendix C.1. Prompts for Summarization and Abstraction

Summarization and abstraction methods rewrite long interaction histories into shorter executable states. Table A1 compares the concrete prompt or response-template designs in ReSum and AgentFold. ReSum exposes summarization as a separate tool invocation that periodically converts a growing trajectory into a compact restartable state. AgentFold, by contrast, embeds context folding into the agent's own structured response format, so that the agent jointly reasons, updates its compressed state, and acts.

 doi:10.20944/preprints202605.2065.v1

Table A1. Prompt and response-template details for summarization and abstraction methods. ReSum uses explicit summarization and summary-conditioned reasoning prompts, whereas AgentFold embeds context folding into the agent's structured response format.

Method	Prompt / Template	Input	Output	Compression-specific instruction
ReSum [17]	Context summarization prompt	The original question and the accumulated conversation history, which may include the agent's previous reasoning, tool calls, observations, and intermediate findings.	A compact summary block that condenses the previous trajectory into a restartable reasoning state.	The summary tool is instructed to extract task-relevant and reliable information from the history, preserve useful evidence and search progress, and avoid unsupported guesses or uncertain inferences.
ReSum [17]	Summary-conditioned reasoning prompt	The original question together with the generated summary from previous exploration.	Continued reasoning and tool use conditioned on the generated summary.	The agent is instructed to treat the summary as condensed prior context, judge whether it is sufficient, and continue collecting evidence if the summary does not yet support a final answer.
AgentFold [18]	Structured agent response template	The task instruction, tool interface, current multi-scale state summaries, and latest high-fidelity interaction.	A structured response containing thinking, a folding directive, a brief explanation, and the next tool call/action.	Context management is made part of the agent's action space. Each step can update the compressed state while still preserving the latest interaction at high fidelity for immediate reasoning.
AgentFold [18]	Folding directive template	Existing state summaries plus the latest interaction.	A JSON-like folding command specifying which span should be folded and what summary should replace it.	The directive supports two modes: granular condensation, which folds a single latest step while preserving useful information, and deep consolidation, which folds several steps into a coarser summary when they complete a subtask and their intermediate details are no longer critical for further task solving.

Table A2 further summarizes the operational flow of the two methods. Although both methods compress trajectories through natural-language abstraction, their control policies differ. ReSum is closer to periodic tool-mediated summarization, while AgentFold is closer to agent-controlled online state maintenance.

 doi:10.20944/preprints202605.2065.v1

Table A2. Operational flow of ReSum and AgentFold. Both compress trajectories into executable summaries, but they differ in whether summarization is invoked as an external tool or internalized as an agent action.

Method	Trigger	Compression process	Resulting state	Main risk
ReSum [17]	Periodic or budget-driven summarization during long-horizon search.	The agent accumulates a ReAct-style trajectory until summarization is invoked. The summarizer rewrites the prior history into a compact state, after which the agent continues from the original question plus the generated summary rather than the full history.	A compact, restartable reasoning state that retains prior discoveries, useful clues, and unresolved directions.	The summary may omit evidence, over-compress uncertainty, or fail to preserve fine-grained dependencies needed in later search.
AgentFold [18]	Agent-initiated folding at each step.	At each step, the agent keeps the most recent interaction as high-fidelity working memory and issues a folding directive that either condenses a single step into a fine-grained summary or consolidates multiple steps into a coarser summary.	A multi-scale memory state consisting of fine-grained local summaries and coarser strategic summaries.	The agent may fold too aggressively, consolidate heterogeneous intermediate steps, or abstract away details that later become necessary.

Appendix C.2. Algorithms for Pruning and Reduction

Pruning and reduction methods differ from abstractive summarization because they preserve selected source content exactly. This property is especially important for coding agents, where identifiers, file paths, syntax, and local dependencies often need exact retention. SWE-Pruner [46] is a representative pruning-based method: it operates as a middleware between the coding agent and the environment, intercepts long code observations, and returns a shorter pruned observation to the agent.

The core scoring and aggregation procedure can be summarized as Algorithm A1. The notation is simplified to emphasize the compression operator rather than implementation-specific training details.

Algorithm A1 Task-aware line-level pruning in SWE-Pruner

Require: Code context C, pruning query or goal hint q, scoring model F_θ, threshold τ
Ensure: Pruned code context $\tilde{C}$
Split C into lines $L = \{l_1, \ldots, l_m\}$
Tokenize C into tokens $X = \{x_1, \ldots, x_n\}$
for each token $x_i \in X$ **do**
    Compute token relevance score $s_i = F_\theta(x_i, C, q)$
end for
for each line $l_j \in L$ **do**
    Let T_j be the set of tokens belonging to line l_j
    Aggregate token scores into a line score:

$$\bar{s}_j = \frac{1}{|T_j|} \sum_{x_i \in T_j} s_i$$

end for
Decode or threshold line scores to obtain keep/drop labels $y_j \in \{0, 1\}$
Construct $\tilde{C}$ by preserving lines with $y_j = 1$
return $\tilde{C}$

Appendix D. Case Studies of Context Compression Failure Modes

This appendix provides three illustrative cases corresponding to the temporal failure taxonomy in Section 5. Each case follows the same pipeline: a task produces an original agent trajectory; the trajectory is compressed into a compact state; and the downstream agent fails after compression. The

 doi:10.20944/preprints202605.2065.v1

cases are constructed to clarify where the earliest causal error arises in the compression pipeline: before compression, during compression, or after compression.

Appendix D.1. F1 Example – Authorization State Dropped

Figure 4a illustrates an F1 failure in a file-management task. The agent is asked to delete all `.pdf` files under `~/downloads`. In the original trajectory, the agent logs in, obtains an `access_token`, attempts to inspect the target directory, and observes a `401 Unauthorized` response. The failed API call is not merely an incidental error: it reveals an operational rule that every subsequent file-system API call must explicitly pass the access token.

The crucial state before compression therefore consists of three linked pieces of information: the concrete token value, the API rule requiring the token, and the cause of the previous `401` error. However, the compression controller makes a wrong pre-compression retention decision. It treats the token, the failed API-call details, and the token-passing rule as disposable low-level execution noise, while preserving only the coarse task objective and a vague note that authorization failed.

The resulting compressed state is insufficient for correct continuation. After compression, the downstream agent still knows that it should delete PDFs under the target directory, but it no longer knows how to perform authorized API calls. It therefore repeats unauthorized calls and wastes steps on unnecessary exploration.

This is an F1 failure because the earliest causal error occurs before the compression operation itself. The problem is not that the summarizer distorted selected content, nor that a saved memory could not be retrieved later. Rather, the controller incorrectly decided that essential operational state was not worth preserving. The repeated `401` errors are downstream symptoms of this pre-compression decision error.

Appendix D.2. F2 Example – Uncertainty Loss

Figure 4b illustrates an F2 failure in a research-agent task. The agent is asked to assess whether Material A shows stable room-temperature superconductivity. The original trajectory contains heterogeneous evidence: one source reports a resistance drop near 295K, another reports no Meissner evidence, a third source reports failed reproduction, and a database entry marks the claim as unverified with conflicting reports. The agent's intermediate conclusion is therefore cautious: the evidence is inconclusive.

Unlike the F1 case, the relevant evidence has been collected and is available to the compression module. The failure occurs during compression. When the trajectory is condensed, the compression module collapses the evidence balance into a simplified statement that Material A shows a 295K transition, suggesting possible room-temperature superconductivity. At the same time, the negative and uncertainty-bearing evidence is weakened or dropped: the lack of Meissner evidence, failed reproduction, unverified database status, and overall inconclusiveness no longer strongly constrain the compressed state.

This is an F2 failure because the compression process changes the epistemic status of the trajectory. The original evidence supports an inconclusive assessment, but the compressed state makes the claim appear more suggestive and better supported than it actually is. The downstream agent then overstates the evidence and produces an overconfident, factually misleading final answer.

The key distinction from F1 is that the information was not excluded by a prior retention decision. Instead, the evidence entered the compression step but was transformed into a less faithful representation. Thus, the earliest causal error lies inside the compression operation itself.

Appendix D.3. F3 Example – Wrong Retrieval

Figure 4c illustrates an F3 failure in a web-research task. The agent must find the CEO of Microsoft in 2024. In the original trajectory, it searches for "Microsoft CEO 2024", opens an official Microsoft source, reads the leadership page, and observes that Satya Nadella was appointed Chief Executive

 doi:10.20944/preprints202605.2065.v1

Officer of Microsoft on February 4, 2014. This correct evidence is then summarized and externalized as memory item `M123`, which stores the official Microsoft leadership evidence for the 2024 CEO.

Unlike the F2 case, the relevant information is not lost or distorted during compression. The compression operation successfully externalizes the correct evidence. However, the compressed active context does not contain the full content of `M123`; it contains the task, an external memory pointer, and an access mechanism indicating that the evidence should be retrieved when needed. The correct memory therefore exists, but it must re-enter the active context through a post-compression access step.

The failure occurs after compression. Instead of directly expanding the correct pointer `M123`, the agent issues the query "Microsoft CEO 2024". The retrieval mechanism returns a different memory item, `M087`, which is topically related but temporally mismatched: it states that Steve Ballmer was Microsoft CEO in 2013 based on an older article. The downstream agent integrates this wrong retrieved chunk into the active context and answers Steve Ballmer.

This is an F3 failure because the correct information has been retained, but it is not correctly accessed at the decision point. The earliest causal error is neither a pre-compression decision error nor an in-compression distortion. Rather, it is a post-compression access failure: the correct memory item `M123` exists, but the retrieval or expansion mechanism surfaces `M087` instead. The final answer is wrong because the agent reasons from the wrong retrieved memory, not because the correct evidence was never stored.

Appendix E. Extended Future Directions

The Compression–Retrieval–Memory Boundary

In current agent systems, compression, retrieval, and external memory are often combined, but they serve different functions and have different recoverability properties. A more principled framework is needed to determine when information should be compressed, when it should be stored externally, and when it should remain in active context.

Learning to Compress Recoverably

Most existing learned policies optimize task success and token efficiency, but these objectives do not directly capture whether compressed information can be reconstructed when needed. Future work should incorporate recoverability and dependency preservation into both training objectives and evaluation.

Compression in Multi-Agent Systems

When context is shared across agents, compression becomes a communication and coordination problem rather than a local memory optimization problem. This setting raises new questions about what should be summarized, what should be shared verbatim, and how compression errors propagate across agents.

End-to-End Training under Context Budgets

Compression is usually treated as a separate module from planning and tool use, even though these decisions are tightly coupled in long-horizon agents. Joint optimization under explicit budget constraints may better align compression behavior with downstream task performance.

Domain-Specific Compression Methods

As shown in § 6, coding, web/GUI, and research agents impose different requirements on structural fidelity, recoverability, and evidence precision. This suggests that domain-tailored compression strategies will be important in addition to general-purpose methods.

 doi:10.20944/preprints202605.2065.v1

Standardized Benchmarks and Reproducibility

Existing work differs widely in agent backbone, task set, budget, and compressed target, which makes direct comparison difficult. Future benchmarks should report not only task success and token savings, but also information density, recoverability, error propagation, and controller overhead under controlled settings.

Concluding Discussion

Agent context compression is no longer just a preprocessing convenience. For long-horizon agents, it is a state-management problem over a changing workspace, not a one-shot prompt-shortening step. This survey structures the area along three axes: targets, mechanisms, and control policies. It also introduces a failure taxonomy and an evaluation framework that goes beyond task success. The key challenge is not only stronger compression, but compression that remains recoverable, faithful, and aligned with long-term agent execution.

 doi:10.20944/preprints202605.2065.v1

Table A3. Full Taxonomy Coordinates. Domain icons: Coding Agents, Web Agents, GUI Interaction Agents, Deep-Search Agents, Research Agents, General Agents.

ID	Method	Author & Year	When	What	How	Domain
1	WebAgent	[74]	System-Controlled	Observation	Summarization & Abstraction	
2	PAL-UI	[51]	Agent-Controlled	Trajectory	Externalization & Retrieval	
3	LCoW	[44]	Learned	Observation	Summarization & Abstraction	
4	Sculptor	[75]	Agent-Controlled	Memory State	Truncation & Masking / Abstraction	
5	SWE-Pruner	[46]	Learned	Observation	Pruning & Reduction	
6	AgentOCR	[21]	Agent-Controlled	Trajectory	Representation-Level	
7	Step-DeepResearch	[48]	System-Controlled	Observation	Externalization & Retrieval	
8	MCP-Bench	[43]	System-Controlled	Observation	Summarization & Abstraction	
9	PAACE	[76]	Learned	Plan & Reasoning	Pruning / Abstraction	
10	WebWeaver	[50]	Agent-Controlled	Memory State	Externalization & Retrieval	
11	The Complexity Trap	[15]	System-Controlled	Observation	Truncation & Masking	
12	WebClipper	[47]	Learned	Trajectory	Pruning & Reduction	
13	AOI	[52]	System-Controlled	Memory State	Summarization & Abstraction	
14	COMPASS	[45]	External Controller	Plan & Reasoning	Summarization & Abstraction	
15	ACC	[77]	External Controller	Memory State	Summarization & Abstraction	
16	BACM	[16]	Agent-Controlled	Trajectory	Summarization & Abstraction	
17	Q-KVComm	[41]	System-Controlled	Representation-Level	Representation-Level	
18	Visual MAD	[22]	System-Controlled	Trajectory	Representation-Level	
19	SCMA	[78]	Learned	Plan & Reasoning	Pruning & Reduction	
20	CSIM	[79]	Learned	Plan & Reasoning	Summarization & Abstraction	
21	ACON	[38]	Learned	Trajectory / Observation	Summarization & Abstraction	
22	SAC	[42]	System-Controlled	Representation-Level	Representation-Level	
23	PoC	[80]	External Controller	Trajectory / Observation	Pruning / Abstraction	
24	SWE-AGILE	[81]	Learned	Plan & Reasoning	Truncation / Abstraction	
25	LoCoEval Framework	[82]	System-Controlled	Memory State	Externalization & Retrieval	
26	LongSeeker	[83]	Agent-Controlled	Trajectory	Pruning / Abstraction / Retrieval	

 doi:10.20944/preprints202605.2065.v1

27	Context-Folding	[84]	Agent-Controlled	Trajectory	Summarization & Abstraction	
28	ContextWeaver	[19]	External Controller	Memory State	Pruning & Reduction	
29	HIAGENT	[85]	Agent-Controlled	Plan & Reasoning	Summarization & Abstraction	
30	ProMem	[54]	Agent-Controlled	Memory State	Summarization & Abstraction	
31	OCR-Memory	[57]	External Controller	Memory State	Externalization & Retrieval	
32	AgentDiet	[16]	System-Controlled	Trajectory	Pruning & Reduction	
33	AgentProg	[86]	System-Controlled	Memory State	Pruning & Reduction	
34	GCC	[20]	Agent-Controlled	Memory State	Externalization & Retrieval	
35	TACO	[87]	Learned	Observation	Pruning & Reduction	
36	CodeOCR	[88]	System-Controlled	Observation	Representation-Level	
37	AttentionRAG	[89]	System-Controlled	Observation	Pruning & Reduction	
38	LongCodeZip	[60]	System-Controlled	Observation	Pruning & Reduction	
39	ReSum	[17]	System-Controlled	Trajectory	Summarization & Abstraction	
40	Focus	[39]	Agent-Controlled	Trajectory	Pruning & Reduction	
41	AgentFold	[18]	Agent-Controlled	Trajectory	Summarization & Abstraction	
42	CAT	[49]	Agent-Controlled	Trajectory	Summarization & Abstraction	
43	MEM1	[40]	Learned	Memory State	Summarization & Abstraction	
44	ContextEvolve	[90]	External Controller	Memory State	Summarization & Abstraction	

Table A4. Open-source methods included in this survey.

Method	Open-source link	Year
SWE-Pruner [46]	https://github.com/Ayanami1314/swe-pruner	2026
Step-DeepResearch [48]	https://github.com/stepfun-ai/StepDeepResearch	2025
MCP-Bench [43]	https://github.com/Accenture/mcp-bench	2025
WebWeaver [50]	https://github.com/Alibaba-NLP/DeepResearch	2025
The Complexity Trap [15]	https://github.com/JetBrains-Research/the-complexity-trap	2025
WebClipper [47]	https://github.com/AQ-MedAI/AntAFu-DeepResearch	2026
ACON [38]	https://github.com/microsoft/acon	2025
SAC [42]	https://github.com/lx-Meteors/SAC	2026
SWE-AGILE [81]	https://github.com/KDEGroup/SWE-AGILE	2026
HIAGENT [85]	https://github.com/HiAgent2024/HiAgent	2025
AgentProg [86]	https://github.com/MobileLLM/AgentProg	2026
TACO [87]	https://github.com/multimodal-art-projection/TACO	2026
CodeOCR [88]	https://github.com/YerbaPage/CodeOCR	2026
LongCodeZip [60]	https://github.com/YerbaPage/LongCodeZip	2025
ReSum [17]	https://github.com/Alibaba-NLP/DeepResearch	2026
AgentFold [18]	https://github.com/Alibaba-NLP/DeepResearch	2025
MEM1 [40]	https://github.com/MIT-MI/MEM1	2025
ContextEvolve [90]	https://anonymous.4open.science/r/ContextEvolve-ACC	2026

 doi:10.20944/preprints202605.2065.v1

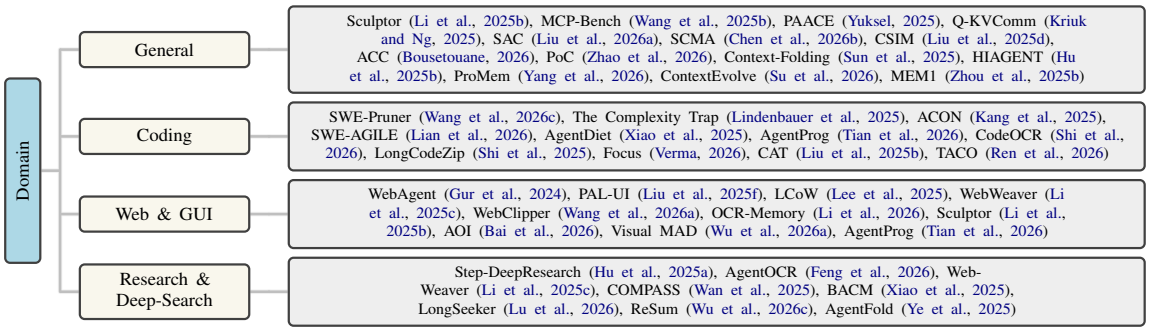

Figure A1. A taxonomy of context management methods from the perspective of domain.

 doi:10.20944/preprints202605.2065.v1

Table A5. Representative datasets and benchmarks for agent context management.

Benchmark	Source & year	Open Source
SWE-bench Verified [61]	ICLR 2024	https://huggingface.co/datasets/SWE-bench/SWE-bench_Verified
GAIA [91]	arXiv 2023	https://huggingface.co/datasets/gaia-benchmark/GAIA
BrowseComp [62]	arXiv 2025	https://github.com/openai/simple-evals
BrowseComp-Plus [63]	ACL 2026	https://huggingface.co/datasets/Tevatron/browsecomp-plus
BrowseComp-ZH [64]	arXiv 2025	https://huggingface.co/datasets/PALIN2018/BrowseComp-ZH
Mind2Web [65]	NeurIPS 2023	https://huggingface.co/datasets/osunlp/Mind2Web
Multimodal-Mind2Web [66]	ICML 2024	https://huggingface.co/datasets/osunlp/Multimodal-Mind2Web
HLE [92]	arXiv 2025	https://huggingface.co/datasets/cais/hle
AppWorld [93]	ACL 2024	https://github.com/StonyBrookNLP/appworld
OfficeBench [94]	arXiv 2024	https://github.com/zlwang-cs/OfficeBench
HotpotQA [95]	EMNLP 2018	https://hotpotqa.github.io/
Natural Questions [96]	TACL 2019	https://github.com/google-research-datasets/natural-questions
TriviaQA [97]	ACL 2017	http://nlp.cs.washington.edu/triviaqa/
GSM8K [98]	arXiv 2021	https://github.com/openai/grade-school-math
MATH / MATH-500 [99]	NeurIPS 2021	https://github.com/hendrycks/math
SWE-bench Lite [61]	ICLR 2024	https://huggingface.co/datasets/princeton-nlp/SWE-bench_Lite
ALFWorld [100]	ICLR 2021	https://github.com/alfworld/alfworld
WebShop [101]	NeurIPS 2022	https://github.com/princeton-nlp/webshop
AndroidWorld / AW-Extend [102]	ICLR 2025	https://github.com/google-research/android_world
DeepResearch Bench [103]	arXiv 2025	https://github.com/Ayanami0730/deep_research_bench
TerminalBench (TB 1.0 / 2.0) [104]	arXiv 2026	https://github.com/harbor-framework/terminal-bench
MRQA [71]	EMNLP WS 2019	https://huggingface.co/datasets/mrqa
CodeXGLUE [105]	NeurIPS / FSE 2021	https://github.com/microsoft/CodeXGLUE
AIOpsLab [106]	MLSys 2025	https://microsoft.github.io/AIOpsLab/
Loghub [72]	ISSRE 2023	https://github.com/logpai/loghub
LongBench [2]	MLSys 2024	https://huggingface.co/datasets/THUDM/LongBench
LongBench V2 [67]	ACL 2025	https://longbench2.github.io
BABILong [68]	NeurIPS 2024	https://github.com/booydar/babilong
ADRS Benchmark [107]	arXiv 2025	https://github.com/UCB-ADRS/ADRS
WorkArena [108]	ICML 2024	https://github.com/ServiceNow/WorkArena
MiniWoB [109]	ICML 2017	https://miniwob.farama.org/
MiniWoB++ [110]	ICLR 2018	https://miniwob.farama.org/
NeedleBench [69]	arXiv 2024	https://github.com/open-compass/opencompass
NIAH [70]	arXiv 2023	https://github.com/open-compass/opencompass
HaluMem [111]	arXiv 2025	https://huggingface.co/datasets/IAAR-Shanghai/HaluMem
LongMemEval [112]	ICLR 2025	https://huggingface.co/datasets/xiaowu0162/longmemeval-cleaned
LoCoEval Framework [113]	arXiv 2026	-
DevEval [114]	ACL Findings 2024	https://github.com/seketeam/DevEval

Continued on next page

 doi:10.20944/preprints202605.2065.v1

Benchmark	Source & year	Open Source
WideSearch [115]	ICLR 2026	https://huggingface.co/datasets/ByteDance-Seed/WideSearch
CompileBench / CRUST-Bench [73]	COLM 2025	https://github.com/QuesmaOrg/CompileBench
GovReport [116]	NAACL 2021	https://gov-report-data.github.io/
SummScreenFD [117]	ACL 2022	https://gov-report-data.github.io/

References

1. Du, Y.; Tian, M.; Ronanki, S.; Rongali, S.; Bodapati, S.B.; Galstyan, A.; Wells, A.; Schwartz, R.; Huerta, E.A.; Peng, H. Context Length Alone Hurts LLM Performance Despite Perfect Retrieval. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2025; Christodoulopoulos, C.; Chakraborty, T.; Rose, C.; Peng, V., Eds., Suzhou, China, 2025; pp. 23281–23298. https://doi.org/10.18653/v1/2025.findings-emnlp.1264.
2. Bai, Y.; Lv, X.; Zhang, J.; Lyu, H.; Tang, J.; Huang, Z.; Du, Z.; Liu, X.; Zeng, A.; Hou, L.; et al. LongBench: A Bilingual, Multitask Benchmark for Long Context Understanding. In Proceedings of the Proceedings of Machine Learning and Systems (MLSys), 2024.
3. Xi, Z.; Chen, W.; Guo, X.; He, W.; Ding, Y.; Hong, B.; Zhang, M.; Wang, J.; Jin, S.; Zhou, E.; et al. The Rise and Potential of Large Language Model Based Agents: A Survey, 2023, [arXiv:cs.AI/2309.07864].
4. Nguyen, D.; Chen, J.; Wang, Y.; Wu, G.; Park, N.; Hu, Z.; Lyu, H.; Wu, J.; Aponte, R.; Xia, Y.; et al. GUI Agents: A Survey. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2025; Che, W.; Nabende, J.; Shutova, E.; Pilehvar, M.T., Eds., Vienna, Austria, 2025; pp. 22522–22538. https://doi.org/10.18653/v1/2025.findings-acl.1158.
5. Zhang, C.; He, S.; Qian, J.; Li, B.; Li, L.; Qin, S.; Kang, Y.; Ma, M.; Liu, G.; Lin, Q.; et al. Large Language Model-Brained GUI Agents: A Survey, 2025, [arXiv:cs.AI/2411.18279].
6. Zhang, W.; Li, X.; Zhang, Y.; Jia, P.; Wang, Y.; Guo, H.; Liu, Y.; Zhao, X. Deep Research: A Survey of Autonomous Research Agents, 2025, [arXiv:cs.IR/2508.12752].
7. Huang, Y.; Chen, Y.; Zhang, H.; Li, K.; Zhou, H.; Fang, M.; Yang, L.; Li, X.; Shang, L.; Xu, S.; et al. Deep Research Agents: A Systematic Examination And Roadmap, 2025, [arXiv:cs.AI/2506.18096].
8. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. ReAct: Synergizing Reasoning and Acting in Language Models, 2023, [arXiv:cs.CL/2210.03629].
9. Liu, N.F.; Lin, K.; Hewitt, J.; Paranjape, A.; Bevilacqua, M.; Petroni, F.; Liang, P. Lost in the Middle: How Language Models Use Long Contexts, 2023, [arXiv:cs.CL/2307.03172].
10. Wang, L.; Ma, C.; Feng, X.; Zhang, Z.; Yang, H.; Zhang, J.; Chen, Z.; Tang, J.; Chen, X.; Lin, Y.; et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science* **2024**, *18*. https://doi.org/10.1007/s11704-024-40231-1.
11. Wang, Y.; Xiong, F.; Wang, Y.; Li, L.; Chu, X.; Zeng, D.D. POSITION BIAS MITIGATES POSITION BIAS: Mitigate Position Bias Through Inter-Position Knowledge Distillation. In Proceedings of the Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing; Christodoulopoulos, C.; Chakraborty, T.; Rose, C.; Peng, V., Eds., Suzhou, China, 2025; pp. 1495–1512. https://doi.org/10.18653/v1/2025.emnlp-main.78.
12. Wang, Y.; Wang, Y.; Yue, Z.; Zeng, H.; Wang, Y.; Lourentzou, I.; Tu, Z.; Chu, X.; McAuley, J. FASA: Frequency-aware Sparse Attention, 2026, [arXiv:cs.CL/2602.03152].
13. Li, H.; Li, Y.; Tian, A.; Tang, T.; Xu, Z.; Chen, X.; Hu, N.; Dong, W.; Li, Q.; Chen, L. A Survey on Large Language Model Acceleration based on KV Cache Management, 2025, [arXiv:cs.AI/2412.19442].
14. Xiao, G.; Tian, Y.; Chen, B.; Han, S.; Lewis, M. Efficient Streaming Language Models with Attention Sinks, 2024, [arXiv:cs.CL/2309.17453].
15. Lindenbauer, T.; Slinko, I.; Felder, L.; Bogomolov, E.; Zharov, Y. The Complexity Trap: Simple Observation Masking Is as Efficient as LLM Summarization for Agent Context Management, 2025, [arXiv:cs.SE/2508.21433].
16. Xiao, Y.A.; Gao, P.; Peng, C.; Xiong, Y. Improving the efficiency of LLM agent systems through trajectory reduction. *arXiv preprint arXiv:2509.23586* **2025**.
17. Wu, X.; Li, K.; Zhao, Y.; Zhang, L.; Ou, L.; Yin, H.; Zhang, Z.; Yu, X.; Zhang, D.; Jiang, Y.; et al. ReSum: Unlocking Long-Horizon Search Intelligence via Context Summarization, 2026, [arXiv:cs.CL/2509.13313].

 doi:10.20944/preprints202605.2065.v1

18. Ye, R.; Zhang, Z.; Li, K.; Yin, H.; Tao, Z.; Zhao, Y.; Su, L.; Zhang, L.; Qiao, Z.; Wang, X.; et al. AgentFold: Long-Horizon Web Agents with Proactive Context Management, 2025, [arXiv:cs.CL/2510.24699].
19. Wu, Y.; Zhang, Y.; Ghosh, S.; Basu, S.; Deoras, A.; Huan, J.; Gupta, G. ContextWeaver: Selective and Dependency-Structured Memory Construction for LLM Agents, 2026, [arXiv:cs.CL/2604.23069].
20. Wu, J.; Hu, M.; Zhu, J.; Pan, J.; Liu, Y.; Xu, M.; Jin, Y. Git Context Controller: Manage the Context of LLM-based Agents like Git, 2026, [arXiv:cs.SE/2508.00031].
21. Feng, L.; Yang, F.; Chen, F.; Cheng, X.; Xu, H.; Wan, Z.; Yan, M.; An, B. AgentOCR: Reimagining Agent History via Optical Self-Compression, 2026, [arXiv:cs.LG/2601.04786].
22. Wu, J.; Sun, Y.; Xie, T.; Chen, S.; Bao, J.; Xu, Y.; Du, G.; Heo, I.; Gutfraind, A.; Wang, X. Cross-Modal Memory Compression for Efficient Multi-Agent Debate, 2026, [arXiv:cs.AI/2602.00454].
23. Mei, L.; Yao, J.; Ge, Y.; Wang, Y.; Bi, B.; Cai, Y.; Liu, J.; Li, M.; Li, Z.Z.; Zhang, D.; et al. A Survey of Context Engineering for Large Language Models, 2025, [arXiv:cs.CL/2507.13334].
24. Jiang, H.; Wu, Q.; Lin, C.Y.; Yang, Y.; Qiu, L. LLMLingua: Compressing Prompts for Accelerated Inference of Large Language Models. In Proceedings of the Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing; Bouamor, H.; Pino, J.; Bali, K., Eds., Singapore, 2023; pp. 13358–13376. https://doi.org/10.18653/v1/2023.emnlp-main.825.
25. Li, Z.; Liu, Y.; Su, Y.; Collier, N. Prompt compression for large language models: A survey. In Proceedings of the Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), 2025, pp. 7182–7195.
26. Li, H.; Li, Y.; Tian, A.; Tang, T.; Xu, Z.; Chen, X.; Hu, N.; Dong, W.; Li, Q.; Chen, L. A survey on large language model acceleration based on kv cache management. *arXiv preprint arXiv:2412.19442* **2024**.
27. Liu, Y.; Fu, J.; Liu, S.; Zou, Y.; Zhang, S.; Zhou, J. KV cache compression for inference efficiency in LLMs: A review. In Proceedings of the Proceedings of the 4th International Conference on Artificial Intelligence and Intelligent Information Processing, 2025, pp. 207–212.
28. Mathur, K.; Mandivarapu, J.K.; Abdi, A.; Chadha, A. Understanding KV Cache Optimization in Large Language Models: A Concise Educational Survey.
29. Zhu, X.; Li, J.; Liu, Y.; Ma, C.; Wang, W. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics* **2024**, *12*, 1556–1577.
30. Liu, D.; Zhu, Y.; Liu, Z.; Liu, Y.; Han, C.; Tian, J.; Li, R.; Yi, W. A survey of model compression techniques: Past, present, and future. *Frontiers in Robotics and AI* **2025**, *12*, 1518965.
31. Xu, C.; McAuley, J. A survey on model compression and acceleration for pretrained language models. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2023, Vol. 37, pp. 10566–10575.
32. Tang, Z.; He, X.; Zhao, T.; Wei, F.; Liu, X.; Dong, P.; Wang, Q.; Li, Q.; Wang, H.; Chen, R.; et al. LLM Agent Memory: A Survey from a Unified Representation–Management Perspective **2026**.
33. Zhang, Z.; Dai, Q.; Bo, X.; Ma, C.; Li, R.; Chen, X.; Zhu, J.; Dong, Z.; Wen, J.R. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems* **2025**, *43*, 1–47.
34. Hu, Y.; Liu, S.; Yue, Y.; Zhang, G.; Liu, B.; Zhu, F.; Lin, J.; Guo, H.; Dou, S.; Xi, Z.; et al. Memory in the age of ai agents. *arXiv preprint arXiv:2512.13564* **2025**.
35. Liu, X.; Yu, H.; Zhang, H.; Xu, Y.; Lei, X.; Lai, H.; Gu, Y.; Ding, H.; Men, K.; Yang, K.; et al. AgentBench: Evaluating LLMs as Agents, 2025, [arXiv:cs.AI/2308.03688].
36. Jiang, H.; Wu, Q.; Luo, X.; Li, D.; Lin, C.Y.; Yang, Y.; Qiu, L. LongLLMLingua: Accelerating and Enhancing LLMs in Long Context Scenarios via Prompt Compression, 2024, [arXiv:cs.CL/2310.06839].
37. Li, Y.; Huang, Y.; Yang, B.; Venkitesh, B.; Locatelli, A.; Ye, H.; Cai, T.; Lewis, P.; Chen, D. SnapKV: LLM Knows What You are Looking for Before Generation. In Proceedings of the Advances in Neural Information Processing Systems; Globerson, A.; Mackey, L.; Belgrave, D.; Fan, A.; Paquet, U.; Tomczak, J.; Zhang, C., Eds. Curran Associates, Inc., 2024, Vol. 37, pp. 22947–22970. https://doi.org/10.52202/079017-0722.
38. Kang, M.; Chen, W.N.; Han, D.; Inan, H.A.; Wutschitz, L.; Chen, Y.; Sim, R.; Rajmohan, S. ACON: Optimizing Context Compression for Long-horizon LLM Agents, 2025, [arXiv:cs.AI/2510.00615].
39. Verma, N. Active Context Compression: Autonomous Memory Management in LLM Agents, 2026, [arXiv:cs.AI/2601.07190].
40. Zhou, Z.; Qu, A.; Wu, Z.; Kim, S.; Prakash, A.; Rus, D.; Zhao, J.; Low, B.K.H.; Liang, P.P. MEM1: Learning to Synergize Memory and Reasoning for Efficient Long-Horizon Agents, 2025, [arXiv:cs.CL/2506.15841].
41. Kriuk, B.; Ng, L. Q-KVComm: Efficient Multi-Agent Communication Via Adaptive KV Cache Compression, 2025, [arXiv:cs.CL/2512.17914].

 doi:10.20944/preprints202605.2065.v1

42. Liu, X.; Zhao, R.; Huang, P.; Liu, X.; Xiao, J.; Xiao, C.; Xiao, T.; Gao, S.; Yu, Z.; Zhu, J. Autoencoding-Free Context Compression for LLMs via Contextual Semantic Anchors, 2026, [arXiv:cs.CL/2510.08907].
43. Wang, Z.; Chang, Q.; Patel, H.; Biju, S.; Wu, C.E.; Liu, Q.; Ding, A.; Rezazadeh, A.; Shah, A.; Bao, Y.; et al. MCP-Bench: Benchmarking Tool-Using LLM Agents with Complex Real-World Tasks via MCP Servers, 2025, [arXiv:cs.CL/2508.20453].
44. Lee, D.; Lee, J.; Kim, K.; Tack, J.; Shin, J.; Teh, Y.W.; Lee, K. Learning to Contextualize Web Pages for Enhanced Decision Making by LLM Agents, 2025, [arXiv:cs.CL/2503.10689].
45. Wan, G.; Ling, M.; Ren, X.; Han, R.; Li, S.; Zhang, Z. COMPASS: Enhancing Agent Long-Horizon Reasoning with Evolving Context, 2025, [arXiv:cs.AI/2510.08790].
46. Wang, Y.; Shi, Y.; Yang, M.; Zhang, R.; He, S.; Lian, H.; Chen, Y.; Ye, S.; Cai, K.; Gu, X. SWE-Pruner: Self-Adaptive Context Pruning for Coding Agents, 2026, [arXiv:cs.SE/2601.16746].
47. Wang, J.; Xie, Z.; Yang, D.; Feng, J.; Shen, Y.; Sun, D.; Long, M.; Jiao, Y.; Tan, Z.; Wang, J.; et al. WebClipper: Efficient Evolution of Web Agents with Graph-based Trajectory Pruning, 2026, [arXiv:cs.AI/2602.12852].
48. Hu, C.; Du, H.; Wang, H.; Lin, L.; Chen, M.; Liu, P.; Miao, R.; Yue, T.; You, W.; Ji, W.; et al. Step-DeepResearch Technical Report, 2025, [arXiv:cs.CL/2512.20491].
49. Liu, S.; Yang, J.; Jiang, B.; Li, Y.; Guo, J.; Liu, X.; Dai, B. Context as a Tool: Context Management for Long-Horizon SWE-Agents, 2025, [arXiv:cs.CL/2512.22087].
50. Li, Z.; Guan, X.; Zhang, B.; Huang, S.; Zhou, H.; Lai, S.; Yan, M.; Jiang, Y.; Xie, P.; Huang, F.; et al. WebWeaver: Structuring Web-Scale Evidence with Dynamic Outlines for Open-Ended Deep Research, 2025, [arXiv:cs.CL/2509.13312].
51. Liu, Z.; Li, J.; Zhao, W.X.; Gao, D.; Li, Y.; rong Wen, J. PAL-UI: Planning with Active Look-back for Vision-Based GUI Agents, 2025, [arXiv:cs.CV/2510.00413].
52. Bai, Z.; Luo, J.; Ni, Z.; Ge, E.; Shi, J.; Zhang, Y.; Gu, J.; Han, Z.; Bao, R.; Hao, J. AOI: Context-Aware Multi-Agent Operations via Dynamic Scheduling and Hierarchical Memory Compression, 2026, [arXiv:cs.MA/2512.13956].
53. Wu, Y.; Zheng, Y.; Xu, T.; Zhang, Z.; Yu, Y.; Zhu, J.; Ma, C.; Lin, B.; Dong, B.; Zhu, H.; et al. ContextBudget: Budget-Aware Context Management for Long-Horizon Search Agents, 2026, [arXiv:cs.AI/2604.01664].
54. Yang, C.; Sun, Z.; Wei, W.; Hu, W. Beyond Static Summarization: Proactive Memory Extraction for LLM Agents, 2026, [arXiv:cs.CL/2601.04463].
55. Liang, S.; Cao, P.; Zhao, J.; Teng, W.; Liao, X.; Zhao, J.; Liu, K. Learning How to Remember: A Meta-Cognitive Management Method for Structured and Transferable Agent Memory, 2026, [arXiv:cs.AI/2601.07470].
56. Chen, D.; Niu, S.; Li, K.; Liu, P.; Zheng, X.; Tang, B.; Li, X.; Xiong, F.; Li, Z. HaluMem: Evaluating Hallucinations in Memory Systems of Agents, 2026, [arXiv:cs.CL/2511.03506].
57. Li, J.; Zhang, Y.; Yang, X.; Qu, J.; Xu, J.; Yang, S.; Ding, J.; Ngai, E.C.H. OCR-Memory: Optical Context Retrieval for Long-Horizon Agent Memory, 2026, [arXiv:cs.CL/2604.26622].
58. Zeng, Y.; Zuo, P.; Lyu, M.; Yang, X.; Wu, H.; Xu, Y.; Yu, Z. KVCache-Centric Memory for LLM Agents, 2026.
59. Wang, L.; Ma, C.; Feng, X.; Zhang, Z.; Yang, H.; Zhang, J.; Chen, Z.; Tang, J.; Chen, X.; Lin, Y.; et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science* **2024**, *18*, 186345.
60. Shi, Y.; Qian, Y.; Zhang, H.; Shen, B.; Gu, X. LongCodeZip: Compress Long Context for Code Language Models, 2025, [arXiv:cs.SE/2510.00446].
61. Jimenez, C.E.; Yang, J.; Wettig, A.; Yao, S.; Pei, K.; Press, O.; Narasimhan, K.R. SWE-bench: Can Language Models Resolve Real-world Github Issues? In Proceedings of the The Twelfth International Conference on Learning Representations, 2024.
62. Wei, J.; Sun, Z.; Papay, S.; McKinney, S.; Han, J.; Fulford, I.; Chung, H.W.; Passos, A.T.; Fedus, W.; Glaese, A. BrowseComp: A Simple Yet Challenging Benchmark for Browsing Agents, 2025, [arXiv:cs.CL/2504.12516].
63. Chen, Z.; Ma, X.; Zhuang, S.; Nie, P.; Zou, K.; Liu, A.; Green, J.; Patel, K.; Meng, R.; Su, M.; et al. BrowseComp-Plus: A More Fair and Transparent Evaluation Benchmark of Deep-Research Agent **2025**. [arXiv:cs.CL/2508.06600].
64. Zhou, P.; Leon, B.; Ying, X.; Zhang, C.; Shao, Y.; Ye, Q.; Chong, D.; Jin, Z.; Xie, C.; Cao, M.; et al. BrowseComp-ZH: Benchmarking Web Browsing Ability of Large Language Models in Chinese, 2025, [arXiv:cs.CL/2504.19314].
65. Deng, X.; Gu, Y.; Zheng, B.; Chen, S.; Stevens, S.; Wang, B.; Sun, H.; Su, Y. Mind2Web: Towards a Generalist Agent for the Web. In Proceedings of the Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS 2023), 2023.

 doi:10.20944/preprints202605.2065.v1

66. Zheng, B.; Gou, B.; Kil, J.; Sun, H.; Su, Y. GPT-4V(ision) is a Generalist Web Agent, if Grounded. In Proceedings of the Forty-first International Conference on Machine Learning, 2024.
67. Bai, Y.; Tu, S.; Zhang, J.; Peng, H.; Wang, X.; Lv, X.; Cao, S.; Xu, J.; Hou, L.; Dong, Y.; et al. LongBench v2: Towards Deeper Understanding and Reasoning on Realistic Long-context Multitasks. In Proceedings of the Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2025.
68. Kuratov, Y.; Bulatov, A.; Anokhin, P.; Rodkin, I.; Sorokin, D.; Sorokin, A.; Burtsev, M. BABILong: Testing the Limits of LLMs with Long Context Reasoning-in-a-Haystack. In Proceedings of the Advances in Neural Information Processing Systems, 2024, Vol. 37, pp. 106519–106554.
69. Li, M.; Zhang, S.; Liu, Y.; Chen, K. NeedleBench: Can LLMs Do Retrieval and Reasoning in 1 Million Context Window?, 2024, [arXiv:cs.CL/2407.11963].
70. Liu, N.F.; Lin, K.; Hewitt, J.; Paranjape, A.; Bevilacqua, M.; Petroni, F.; Liang, P. Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics* **2024**, *12*, 157–173. https://doi.org/10.1162/tacl_a_00638.
71. Fisch, A.; Talmor, A.; Jia, R.; Seo, M.; Choi, E.; Chen, D. MRQA 2019 Shared Task: Evaluating Generalization in Reading Comprehension. In Proceedings of the Proceedings of the 2nd Workshop on Machine Reading for Question Answering, Hong Kong, China, 2019; pp. 1–13. https://doi.org/10.18653/v1/D19-5801.
72. Zhu, J.; He, S.; He, P.; Liu, J.; Lyu, M.R. Loghub: A Large Collection of System Log Datasets for AI-driven Log Analytics. In Proceedings of the IEEE International Symposium on Software Reliability Engineering (ISSRE), 2023. https://doi.org/10.1109/ISSRE59848.2023.00029.
73. Khatry, A.; Zhang, R.; Pan, J.; Wang, Z.; Chen, Q.; Durrett, G.; Dillig, I. CRUST-Bench: A Comprehensive Benchmark for C-to-safe-Rust Transpilation, 2025, [arXiv:cs.SE/2504.15254].
74. Gur, I.; Furuta, H.; Huang, A.; Safdari, M.; Matsuo, Y.; Eck, D.; Faust, A. A Real-World WebAgent with Planning, Long Context Understanding, and Program Synthesis, 2024, [arXiv:cs.LG/2307.12856].
75. Li, M.; Xu, L.H.; Tan, Q.; Ma, L.; Cao, T.; Liu, Y. Sculptor: Empowering LLMs with Cognitive Agency via Active Context Management, 2025, [arXiv:cs.CL/2508.04664].
76. Yuksel, K.A. PAACE: A Plan-Aware Automated Agent Context Engineering Framework, 2025, [arXiv:cs.AI/2512.16970].
77. Bousetouane, F. AI Agents Need Memory Control Over More Context, 2026, [arXiv:q-bio.NC/2601.11653].
78. Chen, Y.; Feng, J.; Yang, W.; Zhong, M.; Shi, Z.; Li, R.; Wei, X.; Gao, Y.; Wu, Y.; Hu, Y.; et al. Self-Compression of Chain-of-Thought via Multi-Agent Reinforcement Learning, 2026, [arXiv:cs.AI/2601.21919].
79. Liu, X.; Li, W.; Sun, W.; Yang, X.; Qin, T.; Gao, X.; Zhou, W. Compressed Step Information Memory for End-to-End Agent Foundation Models. OpenReview (ICLR 2026 Conference Withdrawn Submission), 2025. Withdrawn by authors on 26 Sep 2025.
80. Zhao, R.; Liu, S.; Tang, J.; Liu, L.; Chen, H.; Zhang, W.; Yuan, Y.; Xiao, T.; Zhu, J.; Su, W.; et al. PoC: Performance-oriented Context Compression for Large Language Models via Performance Prediction, 2026, [arXiv:cs.CL/2603.19733].
81. Lian, S.; Liu, J.; Chen, Y.; Chen, Y.; Li, H. SWE-AGILE: A Software Agent Framework for Efficiently Managing Dynamic Reasoning Context, 2026, [arXiv:cs.AI/2604.11716].
82. Liu, Y.; Zhang, L.; Liu, F.; Lin, P.; Li, X. A Scalable Benchmark for Repository-Oriented Long-Horizon Conversational Context Management, 2026, [arXiv:cs.SE/2603.06358].
83. Lu, Y.; Ye, R.; Du, Y.; Wang, J.; Liu, S.; Chen, S. LongSeeker: Elastic Context Orchestration for Long-Horizon Search Agents, 2026, [arXiv:cs.AI/2605.05191].
84. Sun, W.; Lu, M.; Ling, Z.; Liu, K.; Yao, X.; Yang, Y.; Chen, J. Scaling Long-Horizon LLM Agent via Context-Folding, 2025, [arXiv:cs.CL/2510.11967].
85. Hu, M.; Chen, T.; Chen, Q.; Mu, Y.; Shao, W.; Luo, P. HiAgent: Hierarchical Working Memory Management for Solving Long-Horizon Agent Tasks with Large Language Model. In Proceedings of the Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); Che, W.; Nabende, J.; Shutova, E.; Pilehvar, M.T., Eds., Vienna, Austria, 2025; pp. 32779–32798. https://doi.org/10.18653/v1/2025.acl-long.1575.
86. Tian, S.; Wen, H.; Chen, Y.; Liu, J.; Zhao, S.; Liu, G.; Ren, J.; Liu, Y.; Li, Y. AgentProg: Empowering Long-Horizon GUI Agents with Program-Guided Context Management, 2026, [arXiv:cs.AI/2512.10371].
87. Ren, J.; Wu, S.; Li, Y.; Zhu, K.; Xu, S.; Feng, B.; Yuan, R.; Zhang, W.; Batista-Navarro, R.; Yang, J.; et al. A Self-Evolving Framework for Efficient Terminal Agents via Observational Context Compression, 2026, [arXiv:cs.CL/2604.19572].

 doi:10.20944/preprints202605.2065.v1

88. Shi, Y.; Xie, C.; Sun, Z.; Chen, Y.; Zhang, C.; Yun, L.; Wan, C.; Zhang, H.; Lo, D.; Gu, X. CodeOCR: On the Effectiveness of Vision Language Models in Code Understanding, 2026, [arXiv:cs.SE/2602.01785].
89. Fang, Y.; Sun, T.; Shi, Y.; Gu, X. AttentionRAG: Attention-Guided Context Pruning in Retrieval-Augmented Generation, 2025, [arXiv:cs.CL/2503.10720].
90. Su, H.; Zheng, Y.; Li, Y. ContextEvolve: Multi-Agent Context Compression for Systems Code Optimization, 2026, [arXiv:cs.LG/2602.02597].
91. Mialon, G.; Fourrier, C.; Swift, C.; Wolf, T.; LeCun, Y.; Scialom, T. GAIA: a benchmark for General AI Assistants, 2023, [arXiv:cs.CL/2311.12983].
92. Phan, L.; Gatti, A.; Han, Z.; Li, N.; Hu, J.; Zhang, H.; Zhang, C.B.C.; Shaaban, M.; Ling, J.; Shi, S.; et al. Humanity's Last Exam **2025**. [arXiv:cs.LG/2501.14249].
93. Trivedi, H.; Khot, T.; Hartmann, M.; Manku, R.; Dong, V.; Li, E.; Gupta, S.; Sabharwal, A.; Balasubramanian, N. AppWorld: A Controllable World of Apps and People for Benchmarking Interactive Coding Agents. In Proceedings of the Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Bangkok, Thailand, August 2024; pp. 16022–16076. https://doi.org/10.18653/v1/2024.acl-long.850.
94. Wang, Z.; Cui, Y.; Zhong, L.; Zhang, Z.; Yin, D.; Lin, B.Y.; Shang, J. OfficeBench: Benchmarking Language Agents across Multiple Applications for Office Automation, 2024, [arXiv:cs.CL/2407.19056].
95. Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W.W.; Salakhutdinov, R.; Manning, C.D. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In Proceedings of the Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 2018; pp. 2369–2380. https://doi.org/10.18653/v1/D18-1259.
96. Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* **2019**, *7*, 452–466. https://doi.org/10.1162/tacl_a_00276.
97. Joshi, M.; Choi, E.; Weld, D.S.; Zettlemoyer, L. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In Proceedings of the Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, Canada, July 2017; pp. 1601–1611. https://doi.org/10.18653/v1/P17-1147.
98. Cobbe, K.; Kosaraju, V.; Bavarian, M.; Hilton, J.; Nakano, R.; Hesse, C.; Schulman, J. Training Verifiers to Solve Math Word Problems, 2021, [arXiv:cs.LG/2110.14168].
99. Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; Steinhardt, J. Measuring Mathematical Problem Solving With the MATH Dataset. In Proceedings of the Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, 2021.
100. Shridhar, M.; Yuan, X.; Côté, M.A.; Bisk, Y.; Trischler, A.; Hausknecht, M.J. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In Proceedings of the 9th International Conference on Learning Representations, ICLR 2021. OpenReview.net, 2021.
101. Yao, S.; Chen, H.; Yang, J.; Narasimhan, K. WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2022.
102. Rawles, C.; Clinckemaillie, S.; Chang, Y.; Waltz, J.; Lau, G.; Fair, M.; Li, A.; Bishop, W.; Li, W.; Campbell-Ajala, F.; et al. AndroidWorld: A Dynamic Benchmarking Environment for Autonomous Agents, 2024, [arXiv:cs.AI/2405.14573].
103. Du, M.; Xu, B.; Zhu, C.; Wang, X.; Mao, Z. DeepResearch Bench: A Comprehensive Benchmark for Deep Research Agents. *arXiv preprint arXiv:2506.11763* **2025**.
104. Merrill, M.A.; Shaw, A.G.; Carlini, N.; Li, B.; Raj, H.; Bercovich, I.; Shi, L.; Shin, J.Y.; Walshe, T.; Buchanan, E.K.; et al. Terminal-Bench: Benchmarking Agents on Hard, Realistic Tasks in Command Line Interfaces. *arXiv preprint arXiv:2601.11868* **2026**, [arXiv:cs.SE/2601.11868].
105. Lu, S.; Guo, D.; Ren, S.; Huang, J.; Svyatkovskiy, A.; Blanco, A.; Clement, C.B.; Drain, D.; Jiang, D.; Tang, D.; et al. CodeXGLUE: A Machine Learning Benchmark Dataset for Code Understanding and Generation. *arXiv preprint arXiv:2102.04664* **2021**.
106. Chen, Y.; Shetty, M.; Somashekar, G.; Ma, M.; Simmhan, Y.; Mace, J.; Bansal, C.; Wang, R.; Rajmohan, S. AIOpsLab: A Holistic Framework to Evaluate AI Agents for Enabling Autonomous Clouds, 2025.
107. Cheng, A.; Liu, S.; Pan, M.; Li, Z.; Wang, B.; Krentsel, A.; Xia, T.; Cemri, M.; Park, J.; Yang, S.; et al. Barbarians at the Gate: How AI is Upending Systems Research, 2025, [arXiv:cs.OS/2510.06189].

 doi:10.20944/preprints202605.2065.v1

108. Drouin, A.; Gasse, M.; Caccia, M.; Laradji, I.H.; Del Verme, M.; Marty, T.; Boisvert, L.; Thakkar, M.; Cappart, Q.; Vazquez, D.; et al. WorkArena: How Capable Are Web Agents at Solving Common Knowledge Work Tasks? In Proceedings of the Proceedings of the 41st International Conference on Machine Learning (ICML). PMLR, 2024, Vol. 235, pp. 11642–11662.
109. Shi, T.; Karpathy, A.; Fan, L.; Hernandez, J.; Liang, P. World of Bits: An Open-Domain Platform for Web-Based Agents. In Proceedings of the Proceedings of the 34th International Conference on Machine Learning. PMLR, 2017, Vol. 70, pp. 3135–3144.
110. Liu, E.Z.; Guu, K.; Pasupat, P.; Shi, T.; Liang, P. Reinforcement Learning on Web Interfaces using Workflow-Guided Exploration. In Proceedings of the International Conference on Learning Representations (ICLR), 2018.
111. Chen, D.; Niu, S.; Li, K.; Liu, P.; Zheng, X.; Tang, B.; Li, X.; Xiong, F.; Li, Z. HaluMem: Evaluating Hallucinations in Memory Systems of Agents, 2025, [arXiv:cs.CL/2511.03506].
112. Wu, D.; Wang, H.; Yu, W.; Zhang, Y.; Chang, K.W.; Yu, D. LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory. In Proceedings of the The Thirteenth International Conference on Learning Representations (ICLR), 2025.
113. Liu, Y.; Zhang, L.; Liu, F.; Lin, P.; Li, X. A Scalable Benchmark for Repository-Oriented Long-Horizon Conversational Context Management, 2026, [arXiv:cs.SE/2603.06358].
114. Li, J.; Li, G.; Zhao, Y.; Li, Y.; Liu, H.; Zhu, H.; Wang, L.; Liu, K.; Fang, Z.; Wang, L.; et al. DevEval: A Manually-Annotated Code Generation Benchmark Aligned with Real-World Code Repositories. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2024, Bangkok, Thailand, 2024; pp. 3603–3614. https://doi.org/10.18653/v1/2024.findings-acl.214.
115. Wong, R.; Wang, J.; Zhao, J.; Chen, L.; Gao, Y.; Zhang, L.; Zhou, X.; Wang, Z.; Xiang, K.; Zhang, G.; et al. WideSearch: Benchmarking Agentic Broad Info-Seeking, 2025, [arXiv:cs.CL/2508.07999].
116. Huang, L.; Cao, S.; Parulian, N.; Ji, H.; Wang, L. Efficient Attentions for Long Document Summarization. In Proceedings of the Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, 2021; pp. 1419–1436. https://doi.org/10.18653/v1/2021.naacl-main.112.
117. Chen, M.; Chu, Z.; Wiseman, S.; Gimpel, K. SummScreen: A Dataset for Abstractive Screenplay Summarization. In Proceedings of the Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, 2022; pp. 8602–8615. https://doi.org/10.18653/v1/2022.acl-long.589.