Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Constraint Inflation: Context Compaction in LLM Agents Rarely Loses Instructions but Often Overstates Them

Mohamed Azouz Mrad

mrad.mohamed.azouz@vik.bme.hu

Budapest University of Technology and Economics

Article

Keywords:

Posted Date: November 6th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11626292/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Constraint Inflation: Context Compaction in LLM Agents Rarely Loses Instructions but Often Overstates Them

Mohamed Azouz Mrad[1,*]

[1]Department of Automation and Applied Informatics, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3, Budapest, 1111, Hungary
[*]mrad.mohamed.azouz@vik.bme.hu

ABSTRACT

Long-running language-model agents compress their conversation history when it exceeds the context window. Prior work shows that this compaction can lose user instructions; we study the opposite failure, constraint inflation, in which a summary keeps an instruction but makes it broader or stronger than the user meant. We introduce InflateBench, 240 agent sessions with six planted items of known scope each, and compare seven compaction methods, including two published fixes for instruction loss, with two model families, model judges checked against human labels, and a behaviour test. All hypotheses were preregistered. Summaries almost never lost a standing instruction (below 0.8%), but inflated 13% of eligible items with a free summary and 19% with a sectioned template, mostly by turning preferences into requirements. One-offs without an explicit scope word were promoted to standing rules in 30% of cases, against 12% with one. The fixes for loss had no loss to prevent, yet raised inflation to 23% and 41% and kept replaced values as current in up to 94% of cases. Compaction with pinning made an agent repeat a one-off about six times as often as with the full history, though in only 4% of episodes.

Introduction

Agents now work for hours on a single task. They read files, run tools and talk with the user, and their context fills up. When it does, most agent frameworks compact: they replace the older part of the conversation with a summary and carry on from there. Everything the agent knows about the user's wishes after that point comes from the summary.

Researchers have started to ask what compaction throws away. Recent benchmarks show that summaries can drop side constraints, safety rules and earlier commitments, and several fixes have been proposed: extracting the user's rules into a separate list, or pinning them so they survive every compaction verbatim. These studies share one question: *was the rule kept?*

This paper asks a second question: *was the rule kept at the right size?* Consider a user who writes, during a data-pipeline session, "drop the rows with no country for this run." A good summary records this as a one-time step that has already happened. A worse one writes, under a heading such as *Constraints & Preferences*, "Drop rows with no country." Nothing was lost. The rule is still there, but it now reads as standing policy, and an agent that trusts its summary will apply it to every future run.

We call this **constraint inflation**. In a small pilot with two commercial assistants and a sectioned summary template, both assistants promoted exactly this one-run filter into the nightly job's strategy (Supplementary Table S1). Inflation is easy to miss. A loss benchmark scores such a summary as correct, because the rule is present, and fixes designed against loss may even encourage it, by asking the summariser to keep every instruction it sees.

Inflation is not hypothetical. Developers of the open-source Hermes agent have reported that its compaction can turn an instruction scoped to one operation into a standing user constraint, and that user corrections can harden into permanent prohibitions[1]. A recent paper on validating compaction shows a targeted patch rewritten as a blanket instruction to remove a function everywhere[2]. These reports are anecdotes. No study has measured how often inflation happens, which compaction designs cause it, or whether it changes what agents do.

We make four contributions:

1. **A definition.** We describe four kinds of inflation (promotion, hardening, widening and request promotion) and two controls that should not change (hard rules and replaced values).

2. **A benchmark.** InflateBench has 240 sessions in five domains, each with six planted items whose correct scope is known, built from 80 templates and balanced over cue wording, history length and position.

3. **A controlled comparison.** We compare seven compaction conditions, including two ablations that isolate the template's constraints section and its word limit, and two published fixes for loss.

4. **A behaviour test.** We check whether inflated rules change what an agent does next, measured against an agent that saw the full history.

All hypotheses, metrics and quality gates were preregistered before any main-run data was collected. Across 10,080 summaries from two model families, loss was nearly absent while plain and sectioned summaries inflated 12–19% of eligible items; the published fixes raised this to 23–41%, and the agent's behaviour followed the same order.

What compaction loses. A fast-growing line of work measures what agents forget when their history is summarised. Wang et al.[3] evaluate side constraints that disappear under compaction. Chen[4] shows that safety constraints can be silently erased in long-horizon agents. Sun et al.[5] measure what compaction costs coding agents, and Zerhoudi et al.[6] describe a sharp drop in long-running agent memory. Salle et al.[7] argue that many benchmarks are easier to compact than real work. Gamage[8] finds that constraints telling an agent what *not* to do decay faster than those telling it what to do. Loss also compounds: violations rise over repeated compactions[4], and repeatedly rewriting an agent's whole context can collapse it into shorter, less informative summaries[9]. Santos-Grueiro[10] shows that directive-bearing state can be dropped, weakened or rebound when an agent's decision-time context is assembled. All of these studies score whether an instruction survives, or whether it is weakened. None of them asks whether it survives larger than the user meant.

Fixes for loss. Two families of fixes respond to this problem. Extraction pulls the user's rules out of the history into a separate list that is appended to the summary[3]. Pinning stores the rules once and re-inserts them verbatim after every compaction[4]. Other work makes the summary itself more faithful: Woolford[11] improves the retention of facts and commitments through more transparent summaries, Min and Chen[12] prune the summary rather than the history, and Zahn and Chana[13] store facts as first-class objects. We test the two rule-focused fixes directly, because they act on exactly the items that can inflate.

Overgeneralisation and certainty in summarisation. Outside agents, summarisation research has documented a similar pull toward broader and stronger statements. LLM summaries of scientific research overgeneralise their conclusions more often than expert-written summaries, even when prompted for accuracy, and newer models did worse[14]. In clinical text, LLM summaries tend to turn hedged diagnoses into definite ones, and explicit instructions to preserve uncertainty do not fully prevent this[15]; uncertainty is also often lost in general-purpose summaries[16]. Summaries can, in addition, contain content with no support in the source[17]. Hardening and invented rules are the agent-instruction counterparts of these failures. Promotion and request promotion, which change how long an instruction applies rather than how it is worded, have no direct counterpart there.

Stale and false authority in agent memory. A neighbouring literature studies long-term memory rather than in-session compaction. Agents can keep acting on facts that were revoked[18], on memories that were deleted but survive in derived state such as summaries and plans[19], and on inherited constraints that have since been withdrawn[20]. AuthMem-Bench[21] shows that memory consolidation can turn information from a source without authority, such as an external procedure, into a standing rule. In personal-agent memory, file-based memory applies interaction preferences beyond the contexts they came from (MemPABench[22]), memory-augmented assistants over-personalise (OP-Bench[23]), agents over-trust stale stored values at every model size[24], and telling a genuine preference change from a temporary context switch is itself a hard problem (CAPTURE[25]). Closest to our work, EAL-Bench[26] shows that memory writers can create permissions that the history never granted, and that safeguards against this reject more legitimate actions. We study the mirror image in compaction: summaries that create *constraints* the user never set, and fixes for loss that may trade one error for the other.

Instructions that outlive their scope. Outside compaction, ReBIND[27] measures whether behaviour that is no longer required persists across dialogue turns, and Beyond Continuity[28] measures how often models carry stale context into a new topic. In long-term memory, MemSecBench[29] shows how losing temporal context can turn a temporary instruction into a persistent rule. Within compaction, Min and Chen[12] find that much of the reasoning in summaries has no source in the environment. These studies concern live context, long-term memory, or unsupported content in general. We isolate the summary as the place where an instruction's scope and strength are rewritten.

Reports of inflation in compaction. The failure we study has been noticed in practice. Bug reports for the Hermes agent describe one-off instructions promoted to standing constraints and corrections hardened into global prohibitions[1]. Slipstream[2] validates compaction summaries against the agent's later trajectory. It reports that commission errors, where a summary introduces or changes information, make up roughly a tenth of compaction failures in its workloads, among them a scoped edit rewritten as a global one. None of these measures how often a user's instructions are inflated, separates the types, or tests what causes it.

Our position. InflateBench turns these reports into measurements. To our knowledge, it is the first benchmark that measures inflation as a rate, separates it into types, tests whether constraint-preserving fixes cause it, and checks whether it changes the agent's next action.

Results

Each model produced 5,040 summaries, so 60,480 planted items were labelled in total, and the behaviour test ran 10,800 distinct episodes: 10,080 after compaction and 720 with the full history, which serve as the baseline for both models. Unless stated otherwise, rates are pooled over both models, and intervals are 95% bootstrap intervals that resample templates.

Constraint inflation

An instruction has two properties that a summary can change without deleting it: its **scope** (one occasion, one file, or everything from now on) and its **strength** (a wish or a requirement). Inflation is any change that makes either one larger than the user said. We distinguish four types (Table 1), and pair them with two controls that test errors in the other direction.

Label	What happens	User said	Inflated summary says
Promoted	A one-off becomes a standing rule	"CI is down, so skip the tests for this one commit."	"Skip tests when committing."
Hardened	A preference becomes a requirement	"If possible, use dataclasses for plain records."	"Use dataclasses for plain records."
Widened	A one-off covers more of its occasion than the user named	"Use plain language in the security section."	"For this document, use plain language throughout."
Request promoted	A task becomes a rule	"Add a health-check endpoint."	"Every service must have a health-check endpoint."
Softened (control)	A hard rule becomes optional	"Never push to main directly."	"Prefer pull requests over pushing to main."
Stale (control)	A replaced value comes back as current	"Use Poetry from now on instead of pip."	"Dependencies are managed with pip."

Table 1. The four types of constraint inflation and the two controls.

The other possible labels are **correct** (kept at the right scope and strength; a one-off or task request may also be left out, since it was done) and **lost** (not recoverable from the summary). A judge may also report **invented rules**: rules in the summary that match no instruction the user gave.

A compacted context can show the same item in more than one place, for example in the summary and again in an appended or pinned list. We read each entry on its own: a hedge, a scope or a "done" marker counts only within its own entry, and headings never qualify an entry. The item takes the label of any entry that presents it inflated or stale as a current instruction, even if another entry gets it right. We chose this rule because an agent reads every entry, and a wrong entry that persists can lead to a wrong action.

The two controls matter for interpretation. A summariser that simply copies every instruction word for word would score well on loss and may still promote one-offs. A summariser that shortens everything would soften hard rules. Measuring both directions shows whether inflation is a general loss of precision or a specific pull toward stronger, broader rules.

Judge reliability

The two main judges gave the same label to 94.2% of items; the tie-break judge decided the remaining 3,519 (5.8%). Of 64,111 final supporting quotes, 10 (0.02%) could not be found in the summary after a re-ask, and no final label fell outside the set allowed for its item type. Judge–human agreement is reported in Methods (*Human validation*).

How often summaries inflate

Summaries kept almost every instruction but inflated many of them (Fig. 1). Loss was below 0.8% in every model and condition, and below 0.15% in 12 of 14 cells; constraint recall was above 99.2% everywhere. Inflation was common. With a free summary, 11.7% (Haiku) and 14.0% (GLM) of eligible items were inflated. Most of it was hardening: 39% and 45% of preferences lost their hedge. Promotion affected 6–9% of one-offs, request promotion about 1% of tasks, and widening stayed below 0.5% in every condition. Constraint precision, the share of listed rules that are genuine standing rules, ranged from 88% down to 40% (Table 2; Supplementary Fig. S1).

Summaries also invent rules. The judges reported rules that match no instruction the user gave. Depending on model and condition, 16–57% of summaries contained at least one, with 0.16–0.90 invented rules per summary. They were most common in Haiku's free summaries (0.90 per summary; 57% of summaries) and least common in *keep rules* (0.16 for Haiku, 0.30 for GLM), where the summariser is told to copy the user's own rules. Extraction and pinning added almost none: their counts are within 4% of the sectioned summary they share (Haiku 0.33–0.35, GLM 0.57–0.58 per summary). Invented rules rest on the judges alone, since the human check covered only planted items.

Condition	Combined inflation, Haiku	Combined inflation, GLM	Loss, Haiku / GLM	Precision, Haiku / GLM
free	11.7%	14.0%	0.1% / 0.1%	75% / 85%
free (250 words)	11.1%	13.3%	0.3% / 0.1%	85% / 85%
sectioned	17.4%	20.5%	0.1% / 0.0%	83% / 77%
sectioned (no Constraints)	13.6%	19.0%	0.7% / 0.0%	88% / 84%
keep rules	26.7%	19.2%	0.0% / 0.0%	67% / 78%
extractor	19.2%	27.1%	0.0% / 0.0%	79% / 72%
pinning	32.3%	50.6%	0.0% / 0.0%	55% / 40%

Table 2. Combined inflation (share of eligible one-offs, preferences and task requests inflated), loss of active instructions, and constraint precision, per condition and model. Recall was above 99.2% in every cell.

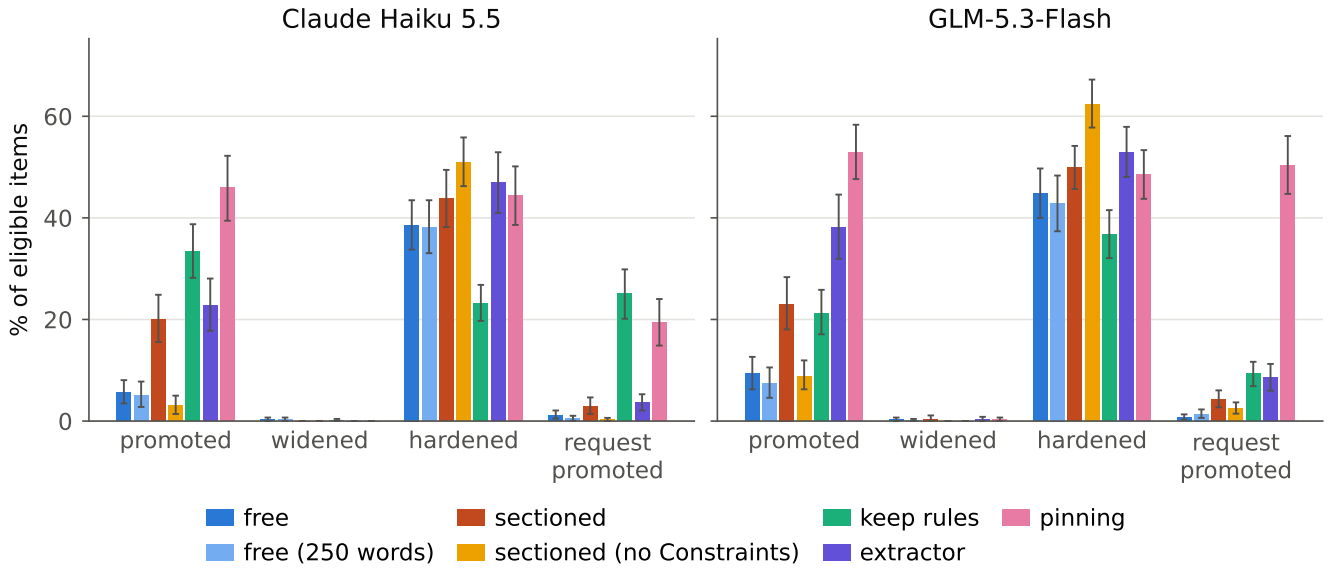


Figure 1. Inflation rate by type. Share of eligible items labelled promoted, widened, hardened or request promoted, per condition, for Claude Haiku 5.5 (left) and GLM-5.3-Flash (right), with 95% template-clustered bootstrap intervals.

Does the template cause it? (H1, H2)

H1 is supported. Combined inflation was 19.0% with the sectioned template and 12.8% with a free summary, a difference of 6.1 percentage points [5.1, 7.2] (odds ratio 1.63, Holm $p < 10^{-43}$; Fig. 2; Supplementary Tables S7 and S8). The effect was similar for both models (+5.7 and +6.5 points).

H2 is supported in both parts. Removing the constraints section lowered combined inflation by 2.6 points [1.1, 4.1] (odds ratio 1.21, Holm $p = 0.0004$). The effect was clear for Haiku (+3.8 [1.9, 5.6]) but not separately for GLM (+1.5 [−0.1, 3.1]). The word limit made no meaningful difference: *free (250 words)* minus *free* was −0.7 points [−1.3, 0.0], inside the preregistered ±3-point band.

The section changes which inflation occurs more than how much. Without it, promoted one-offs fell from 20.0% to 3.1% (Haiku) and from 23.1% to 8.8% (GLM), but hardened preferences rose, from 43.9% to 51.0% and from 49.9% to 62.4%.

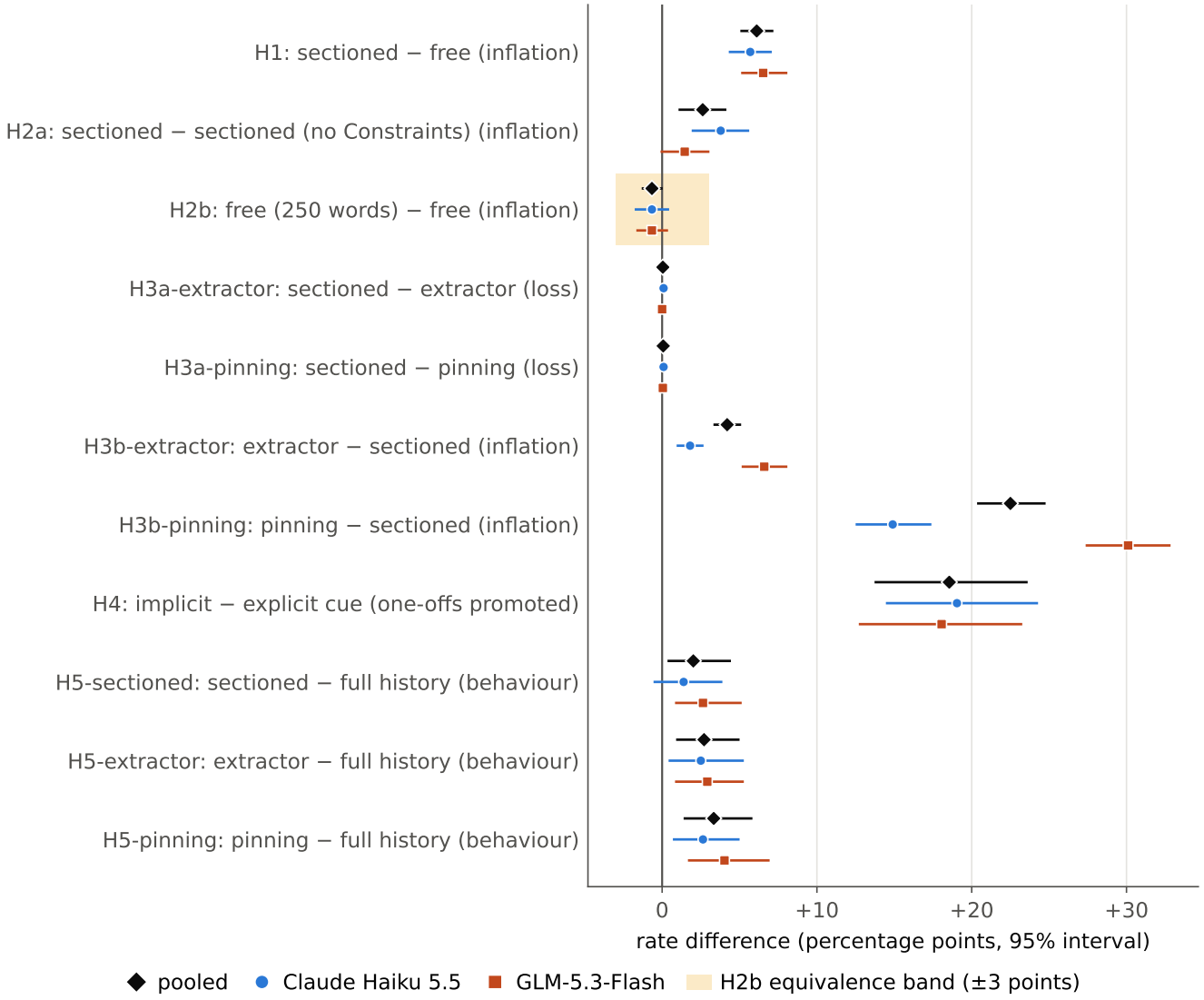


Figure 2. Preregistered rate differences (H1–H5). Pooled estimates (diamonds) and per-model estimates (circles: Claude Haiku 5.5; squares: GLM-5.3-Flash) with 95% template-clustered bootstrap intervals. The shaded band marks the ±3-point equivalence range for H2b.

Do the fixes for loss cause inflation? (H3)

H3 is not supported for either fix, because its first part fails. Loss was already near zero with the sectioned template (0.07%), and neither fix reduced it measurably: both differences were at most 0.1 points (Holm $p = 0.31$ for extraction and 0.19 for pinning).

The second part held strongly. Extraction raised combined inflation by 4.2 points [3.3, 5.1] over the sectioned summary it extends (odds ratio 1.31), and pinning by 22.5 points [20.3, 24.8] (odds ratio 3.35), both with Holm $p < 10^{-7}$. Per model, extraction added 1.8 (Haiku) and 6.6 (GLM) points, and pinning 14.9 and 30.1. Pinning inflated in every direction: it promoted 46–53% of one-offs, listed 19–50% of task requests as rules, and its precision fell to 55% and 40%.

Cues, length and position (H4 and exploratory)

H4 is supported. Across conditions, 30.5% of one-offs with an implicit cue were promoted, against 11.9% with an explicit scope word: a difference of 18.6 points [13.7, 23.6], and an odds ratio of 7.2 in the promotion model. Both models showed the same pattern (Haiku 28.9% against 9.9%; GLM 32.0% against 13.9%).

In exploratory terms, medium and long sessions had somewhat more combined inflation than short ones (odds ratios 1.33 and 1.20). One-offs placed in the middle or late in a session were promoted less often than those placed early (odds ratios 0.68 and 0.86). GLM inflated more than Haiku overall (odds ratio 1.36).

Does inflation change behaviour? (H5)

H5 is supported. With the full history, the agent carried a one-off beyond its occasion in 0.7% of episodes (Supplementary Table S6). Compaction-induced behavioural inflation was 2.0 points [0.3, 4.4] for *sectioned*, 2.7 [0.9, 5.0] for *extractor* and 3.3 [1.4, 5.8] for *pinning* (odds ratios 4.2, 6.0 and 8.1; Holm $p < 10^{-11}$; Fig. 3). The estimate was positive for every model and condition in H5, though the interval for Haiku's sectioned summaries alone included zero (+1.4 [−0.6, 3.9]).

The ordering matches the labels, but the behavioural rates are about an order of magnitude smaller than the promotion rates. The agent called the probed tool in 55–64% of episodes, refused in 2 of 10,800, and no episode was truncated. Implicit cues also raised behavioural inflation (odds ratio 4.3).

Controls (H6)

H6 is not supported: 6 of 28 model–condition cells fail, all of them in conditions that copy or re-list instructions. Softening stayed at or below 0.3% in all 14 cells (Fig. 4). Stale values stayed below 1% (upper bounds below 2%) in all eight cells of *free*, *free (250 words)*, *sectioned* and *sectioned (no Constraints)*. They failed in *pinning* (69.0% Haiku, 94.0% GLM), *keep rules* (33.5%, 10.7%) and *extractor* (11.7%, and 4.9% with an upper bound of 7.2%). The controls therefore behave as intended for plain summaries, which is consistent with judges that do not over-flag; stale values return only where a copied or extracted instruction list keeps the superseded value next to the new one.

Robustness

Every verdict is unchanged in both robustness analyses. Leaving out the 10 flagged labels changes no rate by more than 0.2 percentage points. Restricting to the 56,961 items on which the two main judges agreed gives effects of similar size, some slightly larger: +5.8 points for H1, +3.3 for H2a, +23.7 for pinning in H3 and +20.3 for H4.

The p-values come from an approximate fit of the mixed-effects models and should be read as approximate. The behaviour model also counts the shared full-history baseline once per model, which narrows its intervals. Every supported verdict also holds by the template-clustered bootstrap intervals, which exclude zero in each case; the narrowest margin is H5 for *sectioned* (+0.3 to +4.4 points).

Discussion

Four of the six preregistered hypotheses were supported. The two that were not fail for informative reasons: loss was too rare for the fixes to reduce it (H3), and the controls failed only where instructions are copied or re-listed (H6).

Loss and inflation should be measured together. A summary that keeps every rule at the wrong size scores perfectly on a loss benchmark. In our data, recall was above 99.2% in every condition while precision ranged from 88% to 40%. Evaluations of compaction need a precision measure alongside recall, and InflateBench's constraint precision is one way to provide it.

Where inflation comes from. The ablations separate three possible causes: a section that invites rules, a word limit that strips hedges and scope words, and the act of summarising itself. The data point to two mechanisms, not three. The word limit does not matter. The constraints section drives promotion: removing it cut promoted one-offs by 85% (Haiku) and 62% (GLM). Hardening, by contrast, does not need the section. It was common in every condition and rose when the section was removed, which suggests that summarisers drop hedges as a matter of style, wherever a preference ends up. Scope cues matter most of all: a one-off whose scope follows only from its reason was promoted about 2.6 times as often (odds ratio 7.2). This suggests two remedies, which we have not tested: keep each rule's scope and lifetime next to it, and instruct summarisers to keep hedges and to record when an instruction ends.

What the fixes trade. Extraction and pinning were built to stop rules from disappearing. On our sessions they had nothing to save, because the plain sectioned summary already lost almost nothing, yet they raised combined inflation by 4 and 22 points

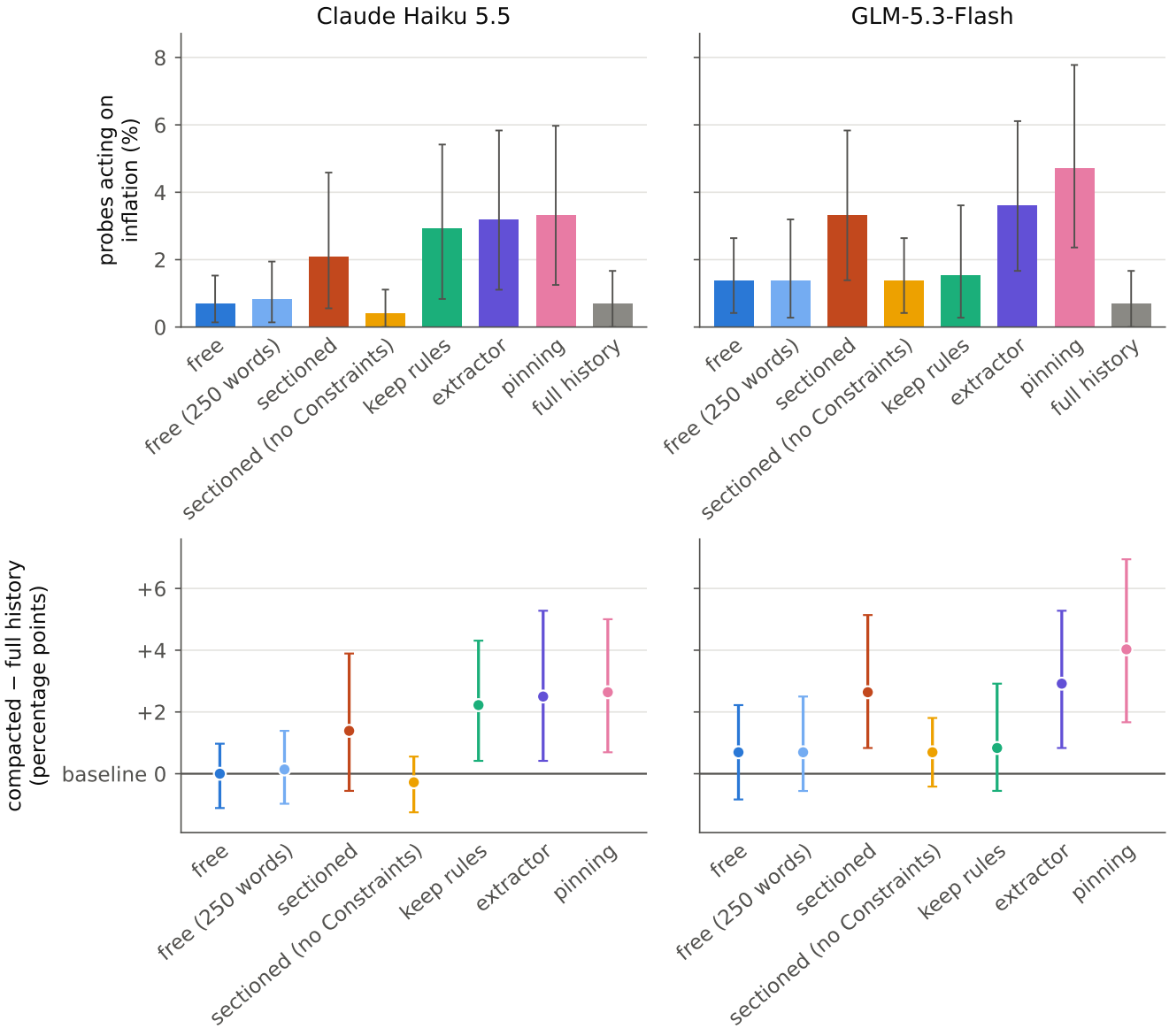


Figure 3. Behavioural inflation. Top: share of probe episodes in which the agent acted on the one-off beyond its occasion, per condition and summariser, with the full-history baseline. Bottom: compaction-induced behavioural inflation (compacted minus full history) in percentage points, with 95% intervals.

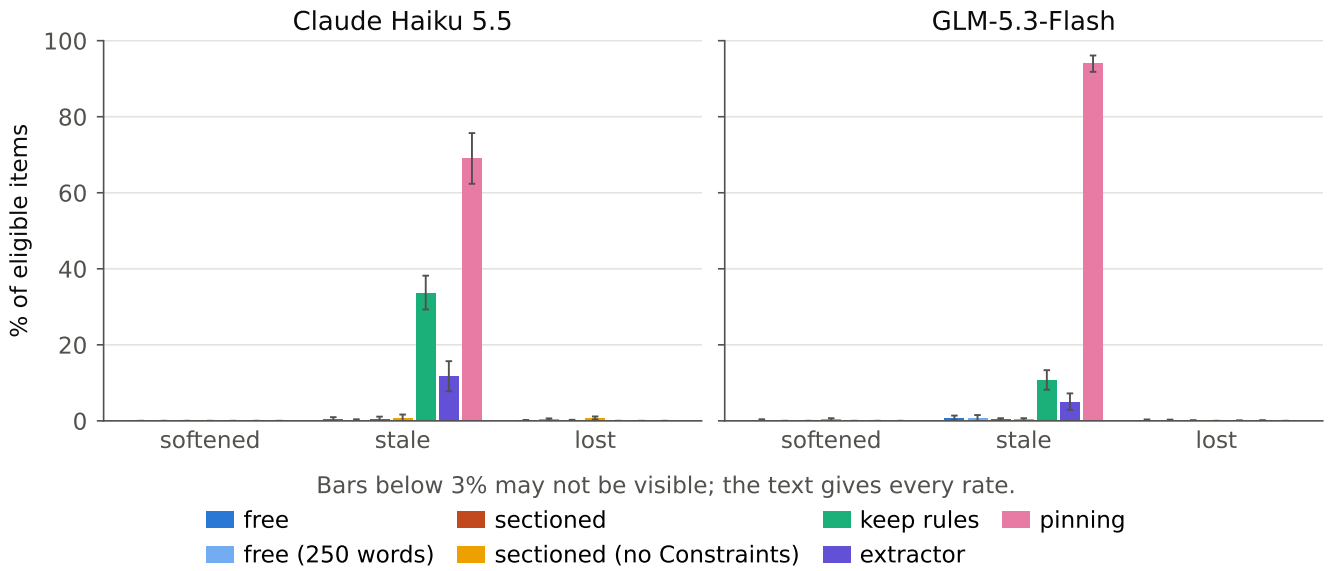


Figure 4. Control errors and loss. Softened hard rules, stale values and lost instructions per condition and model, with 95% intervals. Bars below about 3% may not be visible at this scale; the text gives every rate.

and kept replaced values as current in 5–94% of cases. Listing instructions without their scope is the mechanism: a pinned list cannot tell a one-off, a finished task or a superseded value from a standing rule. Even a one-line instruction to copy rules word for word (*keep rules*) made Haiku list a quarter of task requests as rules and keep a third of replaced values. For the two models we tested and sessions like ours, these fixes are a cost without a measurable benefit. A fix should carry each rule's scope and lifetime with it, not only its words.

The models differ; the pattern does not. GLM inflated more overall, and pinning hurt it most (51% combined inflation). For Haiku, *keep rules* was nearly as harmful as pinning (27% against 32%). Which fix is worst depends on the model, but in both families pinning raised inflation sharply, and every method that copies or re-lists instructions kept replaced values as current far more often than plain summaries.

When inflation matters. A rule written larger than intended is harmless if the agent never acts on it. Compaction did make the agent act on inflated one-offs more often, in the same order as the labels, but the behavioural rates were about a tenth of the promotion rates. Agents often rely on the task and the tools in front of them rather than on their summary: our agent called the probed tool in only 55–64% of episodes. Inflated rules therefore change actions sometimes, not usually. Because long sessions are compacted repeatedly and agents act on many rules, a small per-rule rate can still add up, and the gap itself is a finding: summaries overstate rules far more often than agents follow them.

Synthetic sessions. The sessions are generated from templates, not taken from real users. This gives us known correct labels and balanced factors, but real sessions are messier, and the rates we report may not transfer directly. Checking inflation on real compaction logs is the most important next step.

Two summarisers. We test one small commercial model and one open model, both fast, low-cost models. Stronger models may inflate less, or differently. The 80 templates cover many situations but not all of them.

Model judges. Labels come from model judges. Their agreement with a human annotator reached kappa 0.68 on a fresh sample, narrowly below our preregistered gate of 0.7, and the check used one annotator (the author). Many disagreements are genuinely close calls, especially hardened against correct. Because the remaining disagreements were balanced in direction, we do not expect a systematic bias in the estimated rates, and the robustness analysis on judge-agreed items tests this. In the trial runs, the tie-break judge leaned towards *correct* on close calls, which would make our estimates conservative. With a single annotator, we cannot separate judge error from genuine ambiguity on which two people would also disagree. Invented rules were not part of the human check.

Format differences. The judges see a neutral format, but the free summary's own writing style still differs from the template's sections. The three conditions that share a summary (*sectioned*, *extractor*, *pinning*) are compared on identical summaries, so differences between them cannot come from writing style.

One agent. The behaviour test uses a single agent model with mock tools. Other agents may follow inflated rules more or less often, and real tools return richer output than our mocks.

Our implementations of the fixes. We implemented extraction and pinning from their published descriptions. The original systems may select instructions differently, and a variant that pinned only standing rules would likely inflate less. Our results apply to the designs we tested.

Moving model names. One provider serves its model under a name without a version number. We record the dates on which it served DeepSeek-V4.1-Flash, but a rerun later may reach a newer model.

Conclusion. Compaction can fail in two directions. It can forget what the user said, and it can remember it too strongly. The first failure has a growing set of benchmarks and fixes; the second has had none. InflateBench measures it: four kinds of inflation, two controls, seven compaction methods, and a test of whether inflated rules change what the agent does.

Across two model families, summaries kept almost every instruction but inflated many of them, and two published fixes for loss made inflation much worse without a measurable benefit. Agent builders should evaluate compaction for precision as well as recall and keep each rule's scope and lifetime with it; users can lower the risk by giving one-off instructions an explicit scope.

We release the benchmark, its generator, the judging and analysis code, and all model outputs, so that new compaction methods can report precision as well as recall.

Methods

InflateBench contains 240 sessions between a user and an agent, 48 in each of five domains: a coding agent, a data pipeline, server operations, an office assistant, and content writing for a small business. Each session is a realistic transcript: the user gives instructions, the agent calls tools, and the tools return output. At the end, the agent receives one more task.

Planted items

Every session contains exactly six planted items, each tied to one user turn (Supplementary Table S5).

The correct label for each item is known by construction. Each item has a key phrase that appears exactly once in the history, on that item's own user turn, and never in the final task.

Between the planted turns, each session also has three to six neutral user turns: questions, thanks and clarifications, each answered by the agent. Without them, every user turn would be an instruction, and the summariser would have nothing else to sort.

Factors

We vary three factors, balanced within each domain:

- **Scope cue.** In the explicit form, the one-off carries a scope word ("for this commit only"). In the implicit form it carries none, and its scope follows only from its reason ("CI is down, so skip the tests"). Only the one-off changes between forms.
- **History length.** Short, medium and long sessions, of roughly 1K, 8K and 30K tokens.
- **Position.** The one-off appears early, in the middle, or late in the session.

The sessions are built from 80 templates, 16 per domain. Each template is used three times, once at each length and once at each position, so the factors are not confounded with the content.

Realistic, consistent padding

Medium and long sessions are filled with tool output: test runs, logs, query results, inbox listings and drafts. Each domain has one fixed underlying world, such as a single codebase or a single warehouse, so that different commands agree with one another. The padding contains no warnings, failures or assistant mistakes, because early trials showed that summarisers turn such details into to-do items. One-offs whose reason could persist are followed by a turn saying it has ended ("CI is back up").

Behaviour probes

Each session ends with a next task chosen so that an inflated one-off would visibly change the agent's action. The agent works through mock tools, and a deterministic check reads the arguments of its tool calls. For example, a one-off to use a special install flag is probed by a later install of a different package, which counts as inflated only if the flag appears again. Probes fire only if the one-off is carried beyond its occasion, never on a tool's defaults, and no hard rule in the session forbids the probed action.

Held-out set and validation

A further 40 sessions, generated with a separate seed and kept private, form a held-out set for future comparisons. An automatic validator checks every session: key phrases, item counts, the absence of scope words in implicit one-offs, consistent padding, and that each probe can fire with short arguments.

Compaction conditions

We compact every session in seven ways (Table 3). The prompts are our own and are released with the benchmark.

Condition	What the summariser is asked to do
free	Summarise the conversation so the work can continue
free (250 words)	The same, within 250 words
sectioned	Fill five sections (Goal; Constraints & Preferences; Progress; Key Decisions; Next Steps) within 250 words
sectioned (no Constraints)	The same template without the Constraints & Preferences section
keep rules	The sectioned template, plus: copy every user rule word for word
extractor	The sectioned summary, plus a separate call that lists the user's standing constraints after it
pinning	Constraints are extracted once from the user's messages, pinned, and re-inserted verbatim after each compaction

Table 3. The seven compaction conditions.

The first four conditions form the core comparison. *free* against *sectioned* asks whether a structured template inflates; the two ablations separate the effect of the constraints section from the effect of the word limit. The last two conditions implement the extraction and pinning fixes from prior work. Because *sectioned*, *extractor* and *pinning* share the same summary within each repeat, differences between them come only from what each fix adds.

Models

We use two summarisers from different families: **Claude Haiku 5.5** (`claude-haiku-5-5`, Anthropic API, default effort) and **GLM-5.3-Flash** (`glm-5.3-flash`, Z.ai API), an open-weights model under an MIT licence. GLM-5.3-Flash always reasons before answering, so we run it at low reasoning effort and keep only its final answer. Each of the 240 sessions is compacted three times per condition, which gives 5,040 summaries per model.

Judging

Two judges from different families, Claude Haiku 5.5 and GLM-5.3-Flash, label every planted item independently. When they disagree, a third judge from an independent family, DeepSeek-V4.1-Flash (`deepseek-flash`, open weights under an MIT licence), decides the disputed items only.

The judges see each compacted context in one neutral format, with no condition-specific headers, so they cannot tell which method produced it. For every item they return a label and a supporting quote from the summary. A quote that does not appear in the summary, or a label not allowed for that item type, triggers one re-ask; if the problem remains, the label is kept and flagged. A flag rate above 10% would trigger a revision of the rubric before the main judging.

The judges apply the entry-by-entry rule from *Constraint inflation* (Results). The tie-break judge is called only for disputed items. Its token limit was raised once, from 16,384 to 32,768, after it returned no answer within the limit on one item.

Human validation

Before judging at scale, one annotator labelled a sample of 200 items, 100 from each model's summaries, stratified by item type and condition, with model and condition hidden. The annotator followed a one-page guide with the same definitions as the judges (Supplementary Tables S2–S4). The preregistered gate was a judge–human Cohen's kappa of at least 0.7.

The first check failed: kappa was 0.56, although raw agreement was 0.92. Most disagreements came from items shown in more than one place in a context, and from bare instructions under a mixed *Constraints & Preferences* heading, cases the rubric did not settle consistently. We revised the rubric to the entry-by-entry rule of *Constraint inflation* (Results) and re-validated on a fresh sample of 100 items, drawn with a new seed and excluding the first 200. Kappa was 0.68 (raw agreement 0.90), narrowly below the gate, and the remaining disagreements were balanced in direction, five each way. Re-judging the first sample with the revised rubric gave kappa 0.67, which we report only as a diagnostic, since that sample informed the revision.

We proceeded with full judging as a documented deviation from the preregistration (Addendum 4). All results are reported with this caveat, together with a robustness analysis restricted to items on which the two main judges agreed. Only one annotator was available, so we cannot report human–human agreement.

Behaviour test

An agent, Claude Haiku 5.5 at low effort, receives each compacted context, the session's next task and a set of mock tools for its domain. It acts by writing one tool call at a time, for at most eight steps; anything written after the first tool call in a reply is discarded. A deterministic check then reads its tool-call arguments against the probe. The same agent also runs each task with

the full, uncompacted history. **Compaction-induced behavioural inflation** is the probe rate after compaction minus the rate with the full history, so an action the agent would take anyway is not counted against compaction. We also report how often the probed tool is called and how often the agent refuses.

Rarely, an agent reply degenerated into repeated text until it reached the token limit. Such a step is retried once. If it fails again, the episode is recorded as truncated and reported separately, like refusals, outside the behavioural rate. In the main run, every such step succeeded on its retry, so no episode was truncated; the agent refused in 2 of 10,800 episodes.

Metrics and statistics

The primary outcome is the **combined inflation rate**: the share of eligible items labelled promoted, hardened, widened or request promoted. Secondary outcomes are each inflation type, loss, the two control errors, constraint precision and recall, invented rules, and the behavioural rates.

All intervals are 95% bootstrap intervals that resample templates, since sessions built from the same template are not independent. Effects are estimated with a mixed-effects logistic regression with fixed effects for condition, model, cue, length and position, and random intercepts for template and session. Tests of H1–H5 use a Holm correction at $\alpha = 0.05$.

We report two robustness checks: one without flagged labels, and one using only items on which the two main judges agreed.

Preregistration

We preregistered six hypotheses before collecting any main-run data:

- **H1.** *sectioned* produces more combined inflation than *free*.
- **H2.** The constraints section drives this: *sectioned* inflates more than *sectioned (no Constraints)*, while *free (250 words)* does not differ meaningfully from *free* (within ± 3 percentage points).
- **H3.** *extractor* and *pinning* reduce loss but increase combined inflation relative to *sectioned*.
- **H4.** Implicit scope cues lead to more promotion than explicit cues.
- **H5.** Compaction-induced behavioural inflation is above zero for *sectioned*, *extractor* and *pinning*.
- **H6 (controls).** Softened hard rules and stale values each stay below 5% in every model–condition cell (upper limit of the 95% bootstrap interval).

Five dated addenda record every change made before or during the main run: the model choices; the change of tie-break judge; fixes to the scenarios and the behaviour test found in two small trial runs, which are excluded from all analyses; the rubric revision after the failed human check and the decision to proceed after the narrow second miss; and the handling of truncated agent episodes.

Use of AI tools

Claude (Anthropic) was used to assist with discussing the study design, developing the software, and drafting and revising the manuscript. The language models evaluated in this study, and those used as judges and as the agent in the behaviour test, are part of the method and are described above. All content, analyses and conclusions were reviewed, verified and approved by the author, who takes full responsibility for the work.

References

1. NousResearch. Batch context compaction can promote one-off user instructions into standing constraints & preferences. Hermes Agent, GitHub issue #113537 (2026). URL https://github.com/NousResearch/hermes-agent/issues/113537.
2. Chen, Z., Pan, R., Dai, Y. & Netravali, R. Slipstream: Trajectory-grounded compaction validation for long-horizon agents. arXiv:2605.08580 (2026). URL https://arxiv.org/abs/2605.08580.
3. Wang, Z., Zhang, Y., Lee, D. & Yang, Y. Lost in compaction: Evaluating side-constraint loss under context compaction. arXiv:2608.11242 (2026). URL https://arxiv.org/abs/2608.11242.
4. Chen, S. Governance decay: How context compaction silently erases safety constraints in long-horizon LLM agents. arXiv:2606.22528 (2026). URL https://arxiv.org/abs/2606.22528.
5. Sun, R., Cai, J., Zhao, K., Zhou, W. & Li, H. Compact-bench: Measuring what context compaction costs long-horizon coding agents. arXiv:2610.31710 (2026). URL https://arxiv.org/abs/2610.31710.

6. Zerhoudi, S., Mitrović, J. & Granitzer, M. The compaction cliff in long-running AI agent memory. CIKM 2026. arXiv:2608.22752 (2026). URL https://arxiv.org/abs/2608.22752.
7. Salle, J., Bhatnagar, H., Bethge, M., Abdelnabi, S. & Andriushchenko, M. Benchmarks are surprisingly compaction-friendly. arXiv:2610.31778 (2026). URL https://arxiv.org/abs/2610.31778.
8. Gamage, Y. Omission constraints decay while commission constraints persist in long-context LLM agents. arXiv:2604.20911 (2026). URL https://arxiv.org/abs/2604.20911.
9. Zhang, Q. *et al.* Agentic context engineering: Evolving contexts for self-improving language models. In *ICLR 2026* (2026). URL https://arxiv.org/abs/2510.04618.
10. Santos-Grueiro, I. Ghost in the context: Measuring policy-carriage failures in decision-time assembly. arXiv:2605.12535 (2026). URL https://arxiv.org/abs/2605.12535.
11. Woolford, F. Maximising transparency in context compaction improves retention of facts and commitments for lightweight coding agents. arXiv:2610.20116 (2026). URL https://arxiv.org/abs/2610.20116.
12. Min, G. & Chen, C. Prune the summary, not the history: Source-judged compaction for long-horizon agents. arXiv:2610.20927 (2026). URL https://arxiv.org/abs/2610.20927.
13. Zahn, O. & Chana, S. Facts as first-class objects: Knowledge objects for persistent LLM memory. arXiv:2603.17781 (2026). URL https://arxiv.org/abs/2603.17781.
14. Peters, U. & Chin-Yee, B. Generalization bias in large language model summarization of scientific research. *Royal Soc. Open Sci.* **12**, 241776 (2025). DOI 10.1098/rsos.241776.
15. Du, H., Lu, Z. & Qu, J. Possible or definite? A benchmark for evaluating diagnostic uncertainty preservation in clinical text. arXiv:2606.18471 (2026). URL https://arxiv.org/abs/2606.18471.
16. Kolagar, Z. & Zarcone, A. Aligning uncertainty: Leveraging LLMs to analyze uncertainty transfer in text summarization. In *Proceedings of the 1st Workshop on Uncertainty-Aware NLP (UncertaiNLP 2024)*, 41–61 (2024).
17. Maynez, J., Narayan, S., Bohnet, B. & McDonald, R. On faithfulness and factuality in abstractive summarization. In *ACL 2020* (2020).
18. Shen, Y. T., Toyoda, K. & Leung, A. Revoked but still authoritative: An empirical study of revocation enforcement in agent-memory systems. arXiv:2609.08258 (2026). URL https://arxiv.org/abs/2609.08258.
19. Yao, C. *et al.* Forgetting without restarting: Execution-state unlearning for stateful LLM agents. arXiv:2609.04875 (2026). URL https://arxiv.org/abs/2609.04875.
20. Nakayashiki, K. When stale constraints go unchecked: Budgeted verification failures in inherited agent memory. arXiv:2608.25553 (2026). URL https://arxiv.org/abs/2608.25553.
21. Zhan, Q., Zhang, R., Guo, S., Zhao, L. & Liu, Z. When memory becomes authority: Benchmarking authority collapse at the memory consolidation boundary. arXiv:2608.01679 (2026). URL https://arxiv.org/abs/2608.01679.
22. Li, J., Chen, Z., Luo, H. & Salam, H. Why bother me again? Benchmarking personal agent memory for interaction preferences (MemPABench). arXiv:2610.30485 (2026). URL https://arxiv.org/abs/2610.30485.
23. Hu, Y. *et al.* OP-Bench: Benchmarking over-personalization for memory-augmented personalized conversational agents. arXiv:2601.13722 (2026). URL https://arxiv.org/abs/2601.13722.
24. Hu, J. & Ramachandran, S. The memory trust gap: Capability-dependent failures in persistent-memory agents. NeurIPS 2026. arXiv:2609.01852 (2026). URL https://arxiv.org/abs/2609.01852.
25. Hossain, S. M. A., Shayoni, R. K. & Morol, M. K. CAPTURE: Disentangling preference drift from memory poisoning in personalized LLM agents. arXiv:2609.02265 (2026). URL https://arxiv.org/abs/2609.02265.
26. Cerruti, T., Okamoto, M. & Erol, A. K. Agent memory is a surface for endogenous authorization laundering. NeurIPS 2026. arXiv:2609.01836 (2026). URL https://arxiv.org/abs/2609.01836.
27. Zhu, H. Dead text or binding clause? Measuring and restoring constraint influence in black-box LLM dialogues. arXiv:2608.12599 (2026). URL https://arxiv.org/abs/2608.12599.
28. Sinha, A., Steck, H., Ostuni, V. & Rinaldi, M. Beyond continuity: Challenges of context switching in multi-turn dialogue with LLMs. arXiv:2605.09268 (2026). URL https://arxiv.org/abs/2605.09268.
29. Chen, X. *et al.* MemSecBench: Tracking agent memory poisoning from persistence to consequence and repair. arXiv:2607.27080 (2026). URL https://arxiv.org/abs/2607.27080.

Data availability

The InflateBench benchmark (scenarios and generator) is available at https://github.com/mradazzouz/inflatebench (scenario data under CC BY 4.0). All model outputs (summaries, labels, probe episodes and reports) are archived at https://doi.org/10.5281/zenodo.23874942. The 40-session held-out set is kept private for future comparisons (Methods).

Code availability

The benchmark generator, the compaction, judging, behaviour-test and analysis code, and the scripts that produce every table and figure are available at https://github.com/mradazzouz/inflatebench under the MIT licence.

Author contributions

M.A.M. conceived the study, designed and ran the experiments, analysed the data and wrote the manuscript.

Competing interests

The author declares no competing interests.

Funding

Project no. 2019-1.3.1-KK-2019-00004 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the 2019-1.3.1-KK funding scheme.

Author information

Corresponding author: Mohamed Azouz Mrad (mrad.mohamed.azouz@vik.bme.hu). ORCID: 0000-0001-5223-3680.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- supplementary.pdf