THE RIGHT MEMORY AT THE RIGHT TIME: MEMORY-DRIVEN PROACTIVE ASSISTANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

As assistants move from reactive answering toward proactive intervention, memory changes role: it no longer only supplies the answer to a query, it determines whether and when to speak at all. Prior work in this setting hands the system the relevant fact in context; we study the regime that makes the problem hard, where the evidence justifying an intervention must be mined from a long-horizon personal memory. We introduce the **memory-driven proactive assistance (MPA)** benchmark, pairing egocentric video with year-long, distractor-rich user memories (∼618K tokens on average), human-verified trigger windows, and a protocol that scores content and timing jointly. Every model we evaluate scores near zero, including frontier models given summarized or retrieval-augmented memory. A controlled study explains why: with oracle evidence delivered inside the trigger window a frontier model reaches 96%; letting it choose the timing drops it to 38%, and replacing oracle evidence with retrieval drops it to 15%. Content and timing are separately necessary, and missing either is catastrophic. Guided by this, we build **Gated Proactive Memory (GPM)**, a plug-and-play harness that surfaces memory from a graph-organized store and gates it down to what is worth interrupting for, plus a lightweight RL stage that curbs early and repeated firing. GPM lifts strict accuracy from near zero to 37.0% on Qwen3.8-27B and, with no training at all, 38% on GPT-5.3-Chat; RL post-training reaches 42.8%.[1]

1 INTRODUCTION

As multimodal assistants mature, a shift is underway from a *reactive* paradigm, in which the assistant waits for an explicit query, to a *proactive* one, in which it takes the initiative and engages the user unprompted (Wang et al., 2025a; Thinking Machines Lab, 2026). Consider a user wearing AI glasses: the assistant observes their surroundings in real time and decides for itself when to speak—perhaps to flag a safety hazard, or to remind the user to restock an item. No query is ever issued. Beyond producing an appropriate answer, the assistant must produce it at the right, self-determined moment.

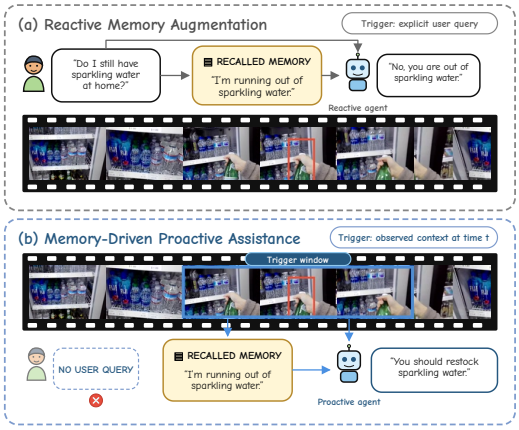


Figure 1: Reactive vs. memory-driven proactive assistance. (a) Memory answers an explicit query; (b) memory is what makes a reminder warranted, and the assistant must also decide *when* to speak.

Central to this shift is the changing role of **memory** (Figure 1). In the reactive paradigm, memory supplies *what to say*: it augments the user's query with related context to ground and personalize the answer (Lewis et al., 2020; Packer et al., 2023). The user asks "do I still have sparkling water at home?" and the assistant retrieves the relevant memory to answer. In the proactive paradigm, memory also determines *when to speak*. Passing sparkling water in a store, the assistant recalls that the user finished the last bottle yesterday, and that recollection is itself what makes a reminder warranted. Memory is no longer only a source of content; it is also the trigger

[1] We will release the MPA benchmark upon publication.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

that governs whether to intervene at all. We call this setting **memory-driven proactive assistance (MPA)**: given a continuous stream of the user's current situation (egocentric video in our benchmark) and a long-term memory of their past, and given no query, the system must decide at each moment when to speak and what to say.

Recent work studies proactive assistance conditioned on user context, but in a regime that sidesteps the central difficulty: the fact justifying an intervention is handed to the model directly, as a short profile (Ran et al., 2026) or a stated goal (Zhang et al., 2025). A deployed assistant faces something different. That fact is one fragment among a year of accumulated experience, and locating it is part of the task. The two regimes differ by orders of magnitude in how much memory must be searched, and performance does not survive the transition. Moreover, success is judged **asymmetrically**. A proactive intervention is an *interruption*: a reminder that arrives at the wrong moment, or that the user never needed, spends attention the user never offered (Deng et al., 2023). Silence is therefore correct almost all of the time, and a response counts only when its content is grounded *and* its timing is right. A restock reminder is unwanted when fired too early (user still in class), too late (shopping done), or repeatedly. This shapes both evaluation and design. Evaluation must credit exactly one well-timed response and must include moments where silence is correct, since otherwise a system that always speaks is indistinguishable from one that speaks well. And the system itself must judge not whether a memory is *relevant*, but whether it *justifies intervening*.

To study this regime we introduce the **MPA benchmark**: egocentric video paired with a controlled, distractor-rich year-long memory of the user's past ($\sim$618K tokens on average) , spanning everyday scenarios (kitchen, shopping, commuting) and reasons to speak up (restocking, safety hazards, unfinished tasks). Each sample is annotated with a human-verified **trigger window** fixing when a response is due and a ground-truth reminder fixing what it should say; a clip counts as correct only when the assistant speaks *once*, inside that window, with grounded content. The benchmark also includes clips on which nothing warrants a response, so abstention is measured rather than assumed. Because the memory is synthesized under our control, we can construct conditions that naturalistic data does not admit, above all supplying a sample's exact justifying evidence.

This enables us to run a controlled study of what the task actually demands: a 2×2 over memory content (oracle evidence vs. retrieval) and timing (given vs. chosen by the assistant; Section 4). Two findings emerge. First, every model we evaluate scores near zero, including frontier models given summarized or retrieval-augmented memory, so the gap is not an artifact of model scale. Second, the two axes are independently necessary: with oracle evidence delivered inside the trigger window a frontier model reaches 96.0%, but only 38.0% if it must choose the timing itself, and 15.3% if the evidence is retrieved rather than given. Generation is not the bottleneck; delivering **the right memory at the right time** is, and dropping either factor is catastrophic.

Guided by this, we design **Gated Proactive Memory (GPM)**, a plug-and-play harness that addresses both axes at once. GPM organizes the user's past into episodic and semantic memories as a graph and traverses it to surface memory related to the current scene; a gate then discards what is accurate but not worth interrupting for, so the generator sees evidence only when a reminder is warranted. A lightweight RL post-training stage adapts the generator to the harness and curbs early and repeated firing. GPM raises strict accuracy from near zero to 37.0% on an open-source model and 38% on a frontier model with no training at all; post-training reaches 42.8%.

In summary, our contributions are as follows:

- We identify **what makes memory-driven proactive assistance hard**. A controlled factorial study separates the problem's two axes and shows they are independently necessary: removing either collapses performance from a 96% oracle ceiling to 38% and 15% respectively.
- We build the instrument that makes this measurable: the **MPA benchmark**, pairing egocentric video with year-long, distractor-rich personal memories and human-verified trigger windows, with a protocol that scores whether, when, and what an assistant says, and credits a system only if it stays silent the rest of the time.
- We show the diagnosis is **actionable**. GPM, a plug-and-play harness, retrieves from a graph-organized memory and gates the result on whether it justifies interrupting—not merely whether it is relevant—lifting strict accuracy from near zero to 37.0% on an open-source model and 38% on a frontier model without any training; a lightweight RL stage reaches 42.8%.

Table 1: Proactive assistance settings, by what the system is given and what the evaluation measures. *Query-free*: no explicit user query at inference. *Personal memory*: kind and typical length of persistent user context. *Retrieval*: relevant evidence must be found rather than supplied in context. *Timing*: response timing is graded. *Content* × *Timing*: the evaluation separates content failures from timing failures. ✓ yes, ○ partially, ✗ no. †Concurrent work (arXiv, July 2026).

Benchmark	Personal memory	Query-free	Retrieval required	Timing graded	Content × Timing
Query-driven streaming					
StreamingBench (Lin et al., 2024)	—	✗	✗	○	✗
OvO-Bench (Niu et al., 2025)	—	✗	✗	○	✗
OmniMMI (Wang et al., 2025d)	—	○	✗	○	✗
ProactiveVideoQA (Wang et al., 2025a)	—	✗	✗	✓	✗
Query-free, with persistent context					
ProAssist (Zhang et al., 2025)	task goal	✓	✗	✓	✗
EgoPro-Bench (Ran et al., 2026)	∼30 tok	✓	✗	✓	✗
EgoServe (Gong et al., 2026)†	video stream	✓	✓	✓	✗
MPA (ours)	**∼618K tok**	✓	✓	✓	✓

2 Related Work

Memory augmentation. Memory augments a model with knowledge beyond its context, retrieved from an external store and, in recent systems, organized as a long-term episodic and semantic record of a user's past (Guu et al., 2020; Karpukhin et al., 2020; Lewis et al., 2020; Izacard & Grave, 2021; Asai et al., 2024; Edge et al., 2024; Packer et al., 2023; Park et al., 2023; Zhong et al., 2024; Huang et al., 2026; Yang et al., 2025; Lin et al., 2025; Long et al., 2025; Yeo et al., 2025; Luo et al., 2025; Jiang et al., 2025). Across this line the query is in hand: memory is retrieved to answer or ground an explicit request, and success is measured by the quality of that answer. In our setting there is no explicit query; the system must continuously interact with its memory as the surroundings unfold and decide whether to intervene at all, so the evidence must be found before it can be used.

Proactive assistance: settings, evaluation, and method. Proactive assistants differ in how much of *what to say* is handed to them in advance (Table 1). In most settings the evidence licensing a response is given, so only the timing is ever in question: prior work supplies an explicit question at fixed timestamps (Lin et al., 2024; Niu et al., 2025; Wang et al., 2025d), a question answered at a model-chosen time (Wang et al., 2025a), a task-guidance goal (Zhang et al., 2025), or a short user profile (Ran et al., 2026)[2] Because the content is given, their evaluation grades mainly timing: needless speech is scored as an aggregate precision term (Zhang et al., 2025; Gong et al., 2026), and hit-style measures credit any interval containing one correct response (Ran et al., 2026). Their methods, in turn, learn when to speak from the supplied content (Chen et al., 2024; Wang et al., 2025b), some with reinforcement learning (Wang et al., 2025c; Ran et al., 2026). Our setting removes this given-content assumption: with no query and the evidence buried in a long personal memory, the system must find *what* to say before it can decide *when*, which forces evaluation to grade content and timing jointly and requires a method built for retrieval rather than given context. Most similar is concurrent work (Gong et al., 2026), which proposes EgoServe, a benchmark for proactive assistance over continuous egocentric video, with the training-free EgoMemo agent. We differ in two ways. At the *memory* level, our controlled memory enables a controlled study that pinpoints why systems fail and shows that success requires the right memory at the right time (Section 4), which a real memory does not permit. At the *method* level, where EgoMemo lets the generator decide at every step whether to intervene and finds most systems perform poorly, we add a gate that filters distracting memory and a post-training stage that largely mitigates this weakness (Section 5).

3 MPA Benchmark

We introduce the **MPA benchmark** for *memory-driven proactive assistance*. Unlike reactive benchmarks that pair an explicit query with a target answer, MPA asks a system to watch an egocentric video stream, consult a long-term memory of the user's past experiences, and emit a grounded proactive intervention *only when warranted* and *only at the right time*. We first formalize the task (Section 3.1) and define the evaluation metrics (Section 3.3); the data construction pipeline is deferred to Appendix C, with human annotation details (annotator count, review questions and response distributions, verdict aggregation, and acceptance rate) in Appendix C.1.

[2] Ran et al. (2026) cap the generated per-video user memory at 10–35 words (their Appendix A.5).

3.1 PROBLEM DEFINITION

Setup. A proactive assistant observes a user's experience as an online multimodal stream $\mathcal{V} = (v_1, v_2, \dots)$, where v_t is the observation at timestep t, and has access to a **long-term memory** $\mathcal{M}$[3] accumulated from the user's past experiences. The setting is *causal* and *query-free*: at each timestep t the system conditions on a recent observation window $\mathcal{V}_{t-n:t} = (v_{t-n}, \dots, v_t)$ together with $\mathcal{M}$, and receives no explicit user query.

Task. We model the system as an always-on causal policy π that at every timestep emits

$$r_t \;=\; \pi(\mathcal{M}, \mathcal{V}_{t-n:t}) \;\in\; \{\varnothing\} \cup \mathcal{A},$$

where $\mathcal{A}$ is the space of natural-language interventions and $\varnothing$ denotes staying silent. The system stays silent most of the time; its meaningful behavior is the *sparse* set of **interventions**

$$E(\pi) \;=\; \big\{\, (t_i, a_i) \;:\; r_{t_i} \neq \varnothing \,\big\},$$

each specifying *what* to say ($a_i \in \mathcal{A}$) and *when* to say it (t_i).

Correctness. The long-term memory determines a set of admissible **trigger windows**

$$\mathcal{W}(\mathcal{M}) \;=\; \{\, W_1, \dots, W_m \,\}, \qquad W_j = [t_j^s, t_j^e],$$

where each window W_j corresponds to one worth-intervening matter, with start and end times t_j^s and t_j^e. If a matter's target appears intermittently (e.g., visible over 2–8 s and again over 40–48 s), W_j is defined by its *first* appearance (2–8 s here): the desired behavior is to intervene at the earliest appropriate moment, and later reappearances do not open new windows: once that earliest appropriate moment has passed, re-raising the same matter is a redundant interruption rather than a new opportunity to help. An intervention $(t_i, a_i) \in E(\pi)$ is **correct** if it fires within its corresponding window ($t_i \in W_j$) and its content is faithful to the relevant memory and grounded in the current frame ($\textsc{Judge}(a_i \mid m_j, v_{t_i}) = 1$, where $m_j \subseteq \mathcal{M}$ is the evidence attached to window j); otherwise it is incorrect. The desired behavior is exactly one correct intervention per window and silence elsewhere: firing outside W_j is early- or late-triggering, and repeated firing is over-triggering.

3.2 DATA STATISTICS

Table 2: Memory history statistics.

Memory history	Mean (min–max)
Time span (days)	362 (330–365)
Events	946 (641–1,020)
Captions	28.1K (19.7K–34.3K)
Tokens (w/ timestamps)	618K (437K–762K)
Target memory tokens	1.2K (0.4K–3.4K)
Target memory share	0.19% (0.06%–0.57%)

Our full benchmark contains 1,804 data pairs, split into 904 for training, 100 for development, and 800 for evaluation; we plan to release the evaluation set. Each pair consists of an egocentric video clip with a single **trigger window**—the interval in which a proactive intervention is warranted—together with a realistic, year-long personal history. The history is a stream of timestamped first-person captions (`[V]` for visible actions, `[C]` for conversational content) that mix main-arc milestones, persona-derived interests, neutral daily distractors, and the evidence that justifies the reminder (see Appendix D.3 for the event types and rendered examples). As summarized in Table 2, each history spans about a year and is long (∼618K tokens), while the target evidence is only ∼0.19% of it, so the relevant evidence is not easy to be found (Appendix D.4). Figure 2 shows the trigger-window and video durations (on average, trigger-window/video ratio ∼0.33); the distributions of scenes and reminder reasons (Figure 6) and representative examples are given in Appendix D. The benchmark also includes 150 negative clips, constructed by removing the target memory from the user history, on which nothing in memory warrants an intervention and the only correct behavior is silence, so any fire is pure cost.

3.3 EVALUATION METRICS

Each instance in our dataset has a single trigger window $W = [t_s, t_e]$, so we state the metrics per clip. Let $\{f_1, \dots, f_k\}$ be the timestamps at which a system fires. Following Section 3.1, a fire is **correct** if it is *in-window* ($f_i \in W$, the right moment) and its content is GOOD, i.e. the intervention a_{f_i} is judged sensible by an **LLM judge** [4] conditioned on the window's evidence $m \subseteq \mathcal{M}$ and the current observation v_{f_i} ($\textsc{Judge}(a_{f_i} \mid m, v_{f_i}) = 1$, the right content).

[3] How a system interacts with $\mathcal{M}$ (how it is organized, and whether it is accessed by, e.g., summarization or retrieval) is itself part of the task and left to the method under evaluation.

[4] The prompt of the LLM judge is shown in Appendix H.

Clip-level behaviors. We then summarize a system's firing behavior on each clip with a few simple measures. A clip is a **HIT** if it has at least one correct (in-window, GOOD) fire, i.e. the system *ever* says the right thing at the right time, ignoring any extra or mistimed fires. We separately flag an **early-trigger** (a fire before the window, $\exists\, f_i < t_s$), a **late-trigger** (a fire after the window, $\exists\, f_i > t_e$), and an **over-trigger** (more than one fire, $k > 1$, i.e. interrupting the user repeatedly). Our primary metric, **PERFECT**, requires the assistant to fire *exactly once*, in-window and GOOD:

$$\text{PERFECT} = \underbrace{(f_1 \in W)}_{\text{right moment}} \wedge \underbrace{\text{GOOD}(a_{f_1})}_{\text{right content}} \wedge \underbrace{(k = 1)}_{\substack{\text{once: no early,} \\ \text{late, or over-trigger}}},$$

that is, a HIT with no early-, late-, or over-trigger. Requiring exactly one intervention is deliberate. Because each utterance is an interruption, repetition is a failure even when the content is right: an assistant that delivers the correct reminder three times has interrupted the user twice for nothing. Unless otherwise noted, we report PERFECT as the main metric and break down the other behaviors above, together with the average number of fires per clip, to show where a system loses performance.

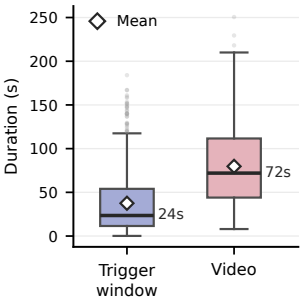


Figure 2: Trigger-window and video durations of the MPA evaluation set.

4 PRELIMINARY FINDINGS

To study how existing proactive implementations behave and where they fall short, we first conduct a preliminary study around two research questions: **RQ1, does having the memory suffice?** We give existing assistants the long-term memory in the most direct ways available: all of it, a summary of it, or retrieved fragments of it, and ask whether any of them yields a usable proactive assistant (Section 4.1). And **RQ2, if not, what is missing — the content, the timing, or both?** We decompose the mechanism into the two things a proactive system must get right, which memory reaches the model and when it reaches it, and vary each independently (Section 4.2). We consider the two paradigms in which such an assistant is commonly deployed. In the **sliding-window** paradigm (Wang et al., 2025a), the model continuously observes the stream through a short sliding window of the most recent frames and decides at each step whether to speak. In the **streaming** paradigm (Chen et al., 2024; Wang et al., 2025c), the model instead processes the stream frame by frame with KV-cache reuse and may emit a response at any frame. Under both paradigms, the agent can see its own past intervention history. Implementation details are in Appendix A.

4.1 RQ1: DOES HAVING THE MEMORY SUFFICE?

We first ask whether simply having the memory is enough. We evaluate a range of open-source and proprietary models under four settings, all letting the model decide its own timing (it may fire at any step): **None-history** (only the current view, no memory), **All-history** (the entire history kept in context at all times; if it exceeds the context window, it is truncated to the most recent portion that fits), **Summary** (a history summary kept in context), and **Naive RAG** (the top-2 retrieved events, injected at every decision frame based on what the agent observes).[5] Results under the windowed paradigm are in Table 3.

Table 3: Straightforward solutions (sliding-window, window size = 4), PERFECT (%); full breakdown in Table 19.

Model	None	All	Summ.	RAG
claude-sonnet-5	0	0	0	8.3
Qwen3.8-27B	0	0	0	4.9
GPT-5.3-Chat	0	0	0	7.8
gemini-3.1-flash-lite	0	0	0	7.8

Across all models and all settings, PERFECT stays low, and the breakdown is telling. Without any history (None-history), no model ever succeeds, confirming that the task genuinely hinges on memory. Yet simply dumping the entire history into the context (All-history) or condensing it into a summary (Summary) fares no better, with PERFECT still at zero for every model: the few memories relevant to the current scene are buried in a long, distractor-rich history and cannot be surfaced by such simple means. Naive RAG helps only marginally: for Qwen3.8-27B it lifts HIT to 36.8% (Table 5), so the right content is sometimes surfaced, yet PERFECT reaches only 4.9%; even frontier models such as claude-sonnet-5 and GPT-5.3-Chat do not exceed 10%, showing that retrieval alone does not solve the joint problem of *when* to speak and *what* to say. The same conclusion holds under

[5]The specific prompts of these settings are shown in Appendix B.

Table 4: Content × timing on Qwen3.8-27B and GPT-5.3-Chat (PERFECT, %; $n = 150$ clips). *Given* supplies evidence only inside the trigger window; *not* leaves timing to the model.

Qwen3.8-27B			**GPT-5.3-Chat**		
	timing given	timing not		timing given	timing not
oracle memory	77.3	26	oracle memory	96	38
naive RAG	7.3	5.3	naive RAG	15.3	8

the streaming paradigm on Qwen3.8-27B (Appendix E, Table 14), confirming that the difficulty is not an artifact of one paradigm.

Observation 1. No. Having the memory does not suffice: with the full history, a summary, or retrieved fragments in context, even frontier models fall far short of a usable assistant.

4.2 RQ2: IF NOT, WHAT IS MISSING — THE CONTENT, THE TIMING, OR BOTH?

We then investigate what a proactive system actually needs for successful MPA, specifically how the memory should be delivered in terms of content and timing. We thus run a controlled study that factors the mechanism into two independent axes on 150 samples: **content**, either an *oracle* memory (the ground-truth evidence) or a *naive RAG* retrieval (usually noisy), and **timing**, either *given* (evidence is provided only inside the trigger window) or *not given* (evidence is available at every step and the model must decide when to fire). Crossing the two axes yields four settings: *oracle + timing given* feeds the ground-truth evidence only inside the trigger window, an upper bound where both content and timing are handed to the model; *oracle + timing not given* keeps the ground-truth evidence visible at every step, so the model must decide when to intervene on its own; *naive RAG + timing given* feeds the retrieved results only inside the trigger window, isolating the effect of noisy content; and *naive RAG + timing not given* retrieves at every step and leaves timing to the model, the fully realistic setting. Table 4 reports PERFECT on the four cells for both Qwen3.8-27B and GPT-5.3-Chat.

Three findings follow. (i) With the right memory at the right time (*oracle + timing given*), the model reaches a high upper bound (**77.3%** for Qwen3.8-27B, **96.0%** for GPT-5.3-Chat); since the appearance of evidence is itself the cue to fire, this cell measures generation rather than decision-making, and it establishes that generation is not the bottleneck. The residual loss is over-triggering inside the (often long) window (Figure 2). (ii) **The right content is not enough** (*oracle + timing not given*): even when the model can always see the oracle memory, leaving the timing to the model collapses PERFECT (77.3→26.0, 96.0→38.0), as it fires at the wrong moments. (iii) **Memory quality is essential** (*oracle* vs. *naive RAG*, under either timing): replacing the oracle memory with naive RAG drops PERFECT to at most 15.3% even when timing is given (77.3→7.3, 96.0→15.3), and further to 5.3% and 8.0% when it is not, because unstructured retrieval feeds the model wrong or noisy evidence. In short, *both* the right memory *and* the right timing are necessary; missing either is catastrophic. The same trend holds under the streaming paradigm (Appendix E, Table 15).

Observation 2. Producing the right *content* is not enough. A proactive system must deliver the *right memory at the right time*: missing either results in catastrophic failure, while with both the generator reaches its ceiling.

5 METHOD

Section 4 shows that the crux of memory-driven proactive assistance is to **deliver the right memory at the right time**: the correct memory surfaced at the wrong moment, or the right timing paired with poorly constructed or irrelevant memory, each collapses performance, and only with both does the generator reach its ceiling. Inspired by this finding, we design **Gated Proactive Memory (GPM)**, a plug-and-play harness that aims to surface the relevant memory precisely when it matters, together with an RL post-training stage that adapts the model to it and further improves performance. We describe the harness in Section 5.1 and the post-training in Section 5.2.

5.1 GATED PROACTIVE MEMORY HARNESS

GPM organizes the user's past into episodic and semantic memories as a graph and traverses it to surface the memories related to the current scene, achieving accurate evidence recall. A gate then filters out irrelevant memories and reorganizes the retained ones into refined evidence, delivering only what is worth intervening on at the right time. Concretely, GPM wraps the generator in a short agentic loop and pairs it with a retrieval module that supplies memory on demand (Figure 3 and Algorithm 1). GPM changes no model weights, so it is plug-and-play across open-source and proprietary models and across both the streaming and windowed paradigms.

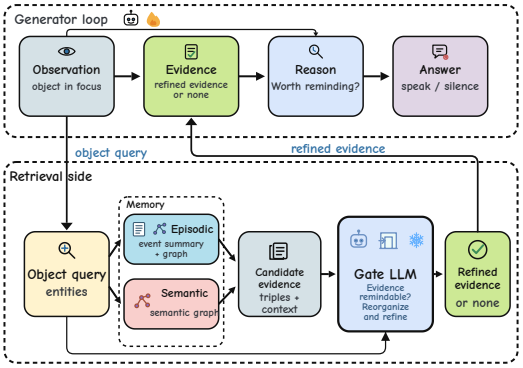


Figure 3: Overview of GPM. The agent proactively observes the current environment and uses the observed entities to retrieve episodic and semantic memories. A gate selects and refines evidence worth surfacing, and the generator combines it with the observation to decide whether and what to say.

Generator loop. At each decision point, the generator follows a two-round agentic loop: *observe* → *retrieve* → *reason* → *answer*. It first identifies the object(s) currently in focus; retrieval then returns refined evidence about those objects or abstains. Conditioned on the evidence and current view, the generator decides whether to speak a grounded reminder or remain silent. The full inputs, serialization tags, and prompts for the windowed and streaming settings are given in Appendix A, Tables 7 and 8.

Memory construction. We organize the user's long-term memory as a graph with two layers: an **episodic** layer of object-keyed memories of past events, so that an object in view directly retrieves related experiences, and a **semantic** layer of standing user facts such as preferences and constraints. Each memory carries an *open-loop* flag marking an unresolved intention, constraint, or issue that may warrant intervention. The construction follows standard KG-memory practice (Gutiérrez et al., 2025; Yeo et al., 2025) (details in Appendix I).

Retrieval and gate. Given the observation entity as a query, we embed-match it to seed entities and pull their one-hop episodic and semantic facts as **candidate** evidence, and run Personalized PageRank on the graph to reach relevant facts beyond one hop. However, many retrieved memories, though real, are not a good reason to remind or intervene: the object may merely co-occur with a past mention, or its open loop may already be resolved. Feeding them all to the model would distract it and cause false triggers. We therefore pass the candidates through the **gate**, a single LLM call that reads them against the observed object, filters out those not worth intervening on, keeps the ones that are, and refines them into a single clear piece of evidence (or abstains); this refined evidence (or nothing) is what the generator consumes. Effectively, the gate asks the question retrieval cannot answer: not whether a memory is relevant, but whether it justifies intervention; removing the gate costs 13.3 points (37.3→24.0, Section 6.3) and GPM as a whole cuts the false-trigger rate on silent clips from 77.3%/84.0% for RAG to 15.3% (windowed/streaming).

5.2 Post-Training

The harness is designed to deliver the right memory at the right time, but it cannot be perfectly accurate. Section 4 shows the generator has a tendency of its own: with correct evidence available at every step, letting it choose the moment still collapses Perfect (77.3→26.0, 96.0→38.0; Table 4). To further lift performance, we post-train the *generator*, teaching it to adapt to the harness and to acquire a sensible firing policy.

Rollout. We adopt the tool-integrated formulation. At a decision step the generator receives the current frame(s) and, following the harness protocol, issues a retrieval call with the observed object; the harness executes retrieval and the gate and returns the refined evidence as a tool turn, after which the generator produces its reason and answer (fire with a grounded reminder, or stay silent). The evidence is injected by the tool rather than generated by the model, so we **mask the evidence tokens** when computing the loss: only the tokens the generator itself produces (its observation call, reason, and answer) contribute to the policy gradient.

Reward. The reward is verifiable from the benchmark annotations. Let $y_t \in \{\text{FIRE}, \text{SILENT}\}$ be the model's action at step t and a the reminder it emits. The gold action g_t depends on both the trigger window and on what the agent has already said: $g_t = \text{FIRE}$ if t lies inside the window *and* the agent has not yet intervened, and $g_t = \text{SILENT}$ otherwise. A repeated in-window intervention is therefore penalized exactly like a mistimed one. We use the unified reward

$$R(y_t, g_t, a) = \begin{cases} +1, & y_t = g_t = \text{SILENT}, \\ +1, & y_t = g_t = \text{FIRE} \ \wedge \ \text{JUDGE}(a) = \text{GOOD}, \\ -1, & \text{otherwise}, \end{cases}$$

where $g_t = \text{FIRE} \iff t \in W \ \wedge \ \nexists\, t' < t$ with $y_{t'} = \text{FIRE}$. Thus, correct silence and a correctly timed, high-quality reminder receive positive credit. The final branch combines every unsuccessful outcome: a miss, a false fire, a low-quality reminder, or a malformed generation. A wrongly timed fire is never sent to the judge. We use a simple binary reward to reduce susceptibility to reward hacking (Wang & Xiong, 2025; Viswanathan et al., 2026). We roll out $G = 4$ trajectories per state and optimize R with GRPO (Shao et al., 2024), which curbs early and repeated firing and yields a further gain on top of the harness. The reward therefore uses the same judge as our metric; we show in Section 6 that the resulting gain is not judge-specific adaptation.

6 Experiment Results

In this section, we conduct comprehensive experiments to evaluate the effectiveness of GPM and its post-training. We first describe the experimental setup, then present the main result, and finally present cross-dataset generalization, negative-sample performance, and an ablation study.

6.1 Experiment Setup

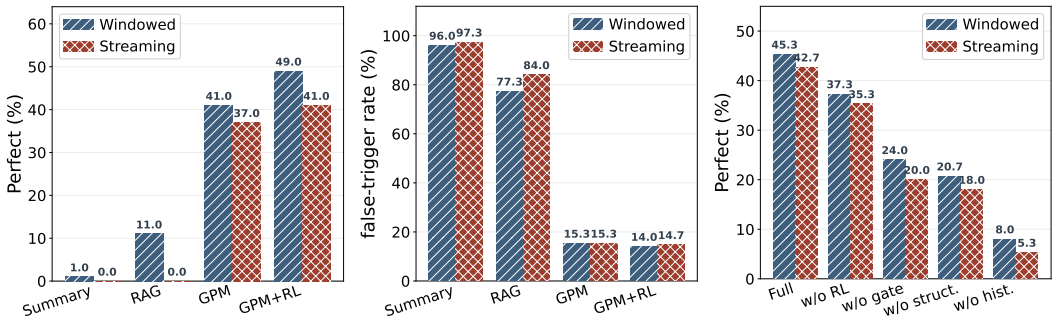
We evaluate GPM and its post-training on MPA under two paradigms: windowed (window size $= 4$ frames) and streaming (up to 30 frames), with prompts and settings in Appendix A. In the windowed setting, Qwen3.8-27B serves as both generator and gate, and we additionally report the GPM harness with GPT-5.3-Chat in both roles. We compare against straightforward memory-augmentation solutions (None-history, Summary, and Naive RAG; defined in Section 4.1) on four backbones: Qwen3.8-27B, GPT-5.3-Chat, claude-sonnet-5, and gemini-3.1-flash-lite. In the streaming setting, Qwen3.8-27B again serves as generator and gate, and we compare against Qwen3.8-27B and two baselines, VideoLLM-online (Chen et al., 2024) and MMDuet2 (Wang et al., 2025c). We use 904 samples for training and 800 for testing (configuration in Appendix G). Unless noted, we report the PERFECT rate with its clip-level breakdown (Section 3.3).

6.2 Main Results

We compare GPM against a range of baselines in Table 5. Two clear trends emerge. First, simply wrapping a base model in the GPM harness—without any training—already improves performance by a large margin: on Qwen3.8-27B, PERFECT rises from at most **8.3%** (the best naive-memory baseline) to **37%** in the windowed setting and from near zero to **33%** in the streaming setting. A similar improvement holds on a frontier model: wrapping GPT-5.3-Chat in GPM (inference-only) reaches **38%** PERFECT (Appendix F, Figure 10). This is because the harness delivers structured, gated memory, surfacing the right evidence only when it is worth intervening, so the generator no longer has to sift the relevant cue from a long, noisy memory on its own. Second, adding RL post-training further lifts PERFECT (windowed 37→**42.8**, streaming 33→**40.1**), with the clearest gains in firing discipline: over-trigger drops markedly (windowed 27.5→18.5, streaming 36.9→27) and so does early-trigger (windowed 17.4→15.3, streaming 17.4→16.3). Because the reward uses the same judge as our metric, one might expect the gain to reflect judge-specific adaptation. It does not. If the policy had learned to satisfy the judge, more of its interventions would be scored GOOD and HIT would rise; instead HIT is flat (61.8 to 61.6). What improves is when and how often the model speaks, over-triggering and early firing, which are checked against the annotated trigger window rather than scored by the judge. This shows that, on top of the harness, the generator better learns to combine the returned evidence with the current observation to judge *whether* and *what* to say, rather than firing prematurely or repeatedly.

Table 5: Performance comparison on MPA under the windowed (top) and streaming (bottom) paradigms (breakdown, %; fires/clip is a count). None- and Summary-history baselines score 0 PERFECT for every backbone (Table 3); full breakdowns are in Appendix J.

	Method	PERFECT ↑	HIT ↑	over-trig ↓	early ↓	late ↓	fires/clip ↓
Windowed paradigm							
baselines	Qwen3.8-27B (RAG)	4.9	36.8	67.6	42.3	28	5.53
	GPT-5.3-Chat (RAG)	7.8	7.8	0	23.1	0	0.39
	claude-sonnet-5 (RAG)	8.3	8.8	0.5	6	1.1	0.18
	gemini-3.1-flash-lite (RAG)	7.8	9.3	4.9	47.8	4.4	1.79
ours	GPM harness (Qwen3.8-27B)	37	**61.8**	27.5	17.4	2.4	1.19
	+ GRPO	**42.8**	61.6	**18.5**	**15.3**	**2**	**1.02**
Streaming paradigm							
baselines	Qwen3.8-27B (RAG)	0.5	2	82	80	62	58
	VideoLLM-online (RAG)	1.6	4.9	62.1	42.3	34.1	49.72
	MMDuet2 (RAG)	0	19.8	72	15.9	48.9	48.23
ours	GPM harness (Qwen3.8-27B)	33	**66.4**	36.9	17.4	3	1.50
	+ GRPO	**40.1**	64.9	**27**	**16.3**	**2.8**	**1.32**



(a) Generalization (Epic-Kitchens) (b) Negative sample (false-trigger ↓) (c) Ablation (150 samples)

Figure 4: Further probing of GPM (Qwen3.8-27B, windowed vs. streaming): (a) zero-shot generalization to Epic-Kitchens (PERFECT, %); (b) false-trigger rate on negative clips (%, lower is better); (c) ablation.

6.3 FURTHER PROBING

To better understand GPM, we probe its generalization, its behavior on negative samples, and the contribution of each component (Figure 4). *Generalization.* Transferred without adaptation to 100 samples from the unseen Epic-Kitchens dataset, the harness lifts PERFECT from at most 11.0 (always-on RAG) to 41.0 windowed and 37.0 streaming, and the Ego4D-trained generator further reaches 49.0 and 41.0, showing that the learned firing policy transfers across domains. *Negative samples.* On 150 clips whose relevant memory is removed, so that the assistant should stay silent throughout, naive methods still fire on most clips (77.3%/84.0% for always-on RAG, windowed/streaming), whereas GPM fires on only 15.3%, and RL lowers this further to 14.0%/14.7%. *Ablation.* On 150 samples randomly chosen from the test set, starting from the harness without RL (37.3 windowed), removing the gate costs 13.3 points, confirming its role in delivering memory at the right time; replacing the structured graph memory with naive RAG, even with the gate kept, costs 16.6 points, confirming the need for the right memory; and hiding the model's own past responses causes the largest drop (to 8.0), showing that the speaking history is key to curbing over-triggering. Streaming shows the same trends.

7 CONCLUSIONS

We studied memory-driven proactive assistance, where memory determines not only what an assistant says but whether and when it should speak at all. A controlled study isolates what the task demands: the right memory and the right timing are independently necessary, and missing either collapses performance from a 96% oracle ceiling to under 40%. Acting on this diagnosis, our gated harness GPM lifts strict accuracy from near zero to 37.0% with no change to model weights, and to 42.8% with lightweight RL post-training.

AI USE STATEMENT

In this work, we used generative AI tools to generate synthetic data sets: the MPA construction pipeline uses LLMs to propose memory evidence, synthesize owner personas and year-long event timelines, and render events into caption streams (Appendix C). All generated samples passed model-based quality control and independent review by 32 trained annotators, and only human-accepted samples are included in the benchmark (Appendix C.1). We have not used generative AI tools for the other tasks with required disclosure. Additionally, we used generative AI tools to polish the writing of this paper, including grammar correction and improvements to clarity and readability. The authors carefully reviewed all AI-assisted edits to ensure they preserved the intended technical meaning and accurately represented our methods and findings. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The benchmark construction pipeline and the human verification protocol are described in Section 3 and Appendix C. Memory construction for the GPM harness is detailed in Appendix I; the windowed and streaming implementations, including the shared retrieval gate prompt, in Appendix A; and the prompts for the naive baselines and the LLM judge in Appendices B and H. Post-training data construction and hyperparameters are given in Appendix G. We will release the benchmark and the evaluation protocol upon publication.

ETHICS STATEMENT

All personal histories in MPA are synthesized and do not correspond to real individuals, so no private information about the recorded participants is introduced. Human verification was performed by 32 trained annotators, who reviewed and validated the trigger windows and rationales for the included samples. More broadly, an assistant that continuously observes a user's surroundings and retains a long-term record of their activity raises privacy considerations that any deployment would need to address; our benchmark studies when such a system should speak, and the interruption cost it imposes when it speaks wrongly, rather than endorsing always-on capture.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations (ICLR)*, 2024.

Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. VideoLLM-online: Online video large language model for streaming video. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The EPIC-KITCHENS dataset. In *European Conference on Computer Vision (ECCV)*, 2018.

Yang Deng, Wenqiang Lei, Minlie Huang, and Tat-Seng Chua. A survey on proactive dialogue systems: Problems, methods, and prospects. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. From local to global: A graph RAG approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

Sitong Gong, Tianyu Yan, Caixin Kang, et al. Vinci2: Providing proactive assistance in continuous egocentric videos. *arXiv preprint arXiv:2607.11523*, 2026.

Kristen Grauman, Andrew Westbury, Eugene Byrne, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. From RAG to memory: Non-parametric continual learning for large language models. *arXiv preprint arXiv:2502.14802*, 2025.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. REALM: Retrieval-augmented language model pre-training. In *International Conference on Machine Learning (ICML)*, 2020.

Wei-Chieh Huang, Weizhi Zhang, Yueqing Liang, et al. Rethinking memory mechanisms of foundation agents in the second half: A survey. *arXiv preprint arXiv:2602.06052*, 2026.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021.

Hongda Jiang, Xinyuan Zhang, Siddhant Garg, et al. Memory-QA: Answering recall questions based on multimodal memories. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Junming Lin, Zheng Fang, Chi Chen, Zihao Wan, Fuwen Luo, Peng Li, Yang Liu, and Maosong Sun. StreamingBench: Assessing the gap for MLLMs to achieve streaming video understanding. *arXiv preprint arXiv:2411.03628*, 2024.

Yueqian Lin, Jingyang Zhang, Qinsi Wang, et al. HippoMM: Hippocampal-inspired multimodal memory for long audiovisual event understanding. *arXiv preprint arXiv:2504.10739*, 2025.

Lin Long, Yichen He, Wentao Ye, et al. Seeing, listening, remembering, and reasoning: A multimodal agent with long-term memory. *arXiv preprint arXiv:2508.09736*, 2025.

Yongdong Luo, Xiawu Zheng, Guilin Li, et al. Video-RAG: Visually-aligned retrieval-augmented long video comprehension. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Junbo Niu, Yifei Li, Ziyang Miao, Chunjiang Ge, Yuanhang Zhou, Qihao He, Xiaoyi Dong, Haodong Duan, Shuangrui Ding, Rui Qian, et al. OvO-Bench: How far is your video-LLMs from real-world online video understanding? In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *ACM Symposium on User Interface Software and Technology (UIST)*, 2023.

Dongchuan Ran, Linyu Ou, Xueheng Li, Wenwen Tong, Chenxu Guo, Hewei Guo, Kaibing Wang, and Lewei Lu. Egopro-bench: Benchmarking personalized proactive interaction in egocentric video streams, 2026. URL https://arxiv.org/abs/2605.07299.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Thinking Machines Lab. Interaction models: A scalable approach to human-ai collaboration. *Thinking Machines Lab: Connectionism*, May 2026. doi: 10.64434/tml.20260511. https://thinkingmachines.ai/blog/interaction-models/.

Vijay Viswanathan, Shiqi Wang, Devamanyu Hazarika, Chirag Nagpal, Tongshuang Wu, Graham Neubig, and Yuning Mao. Discretizing reward models, 2026. URL https://arxiv.org/abs/2606.21795.

Tevin Wang and Chenyan Xiong. Autorule: Reasoning chain-of-thought extracted rule-based rewards improve preference learning, 2025. URL https://arxiv.org/abs/2506.15651.

Yueqian Wang, Xiaojun Meng, Yifan Wang, Huishuai Zhang, and Dongyan Zhao. ProactiveVideoQA: A comprehensive benchmark evaluating proactive interactions in video large language models. *arXiv preprint arXiv:2507.09313*, 2025a.

Yueqian Wang, Xiaojun Meng, Yuxuan Wang, et al. VideoLLM knows when to speak: Enhancing time-sensitive video comprehension with video-text duet interaction format. In *Findings of the Association for Computational Linguistics: EMNLP*, 2025b.

Yueqian Wang et al. MMDuet2: Enhancing proactive interaction of video MLLMs with multi-turn reinforcement learning. *arXiv preprint arXiv:2512.06810*, 2025c.

Yuxuan Wang, Yueqian Wang, Bo Chen, Tong Wu, Dongyan Zhao, and Zilong Zheng. OmniMMI: A comprehensive multi-modal interaction benchmark in streaming video contexts. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025d.

Jingkang Yang, Shuai Liu, Hongming Guo, et al. EgoLife: Towards egocentric life assistant. *arXiv preprint arXiv:2503.03803*, 2025.

Woongyeong Yeo, Kangsan Kim, Jaehong Yoon, and Sung Ju Hwang. WorldMM: Dynamic multimodal memory agent for long video reasoning. *arXiv preprint arXiv:2512.02425*, 2025.

Yichi Zhang, Xin Luna Dong, Zhaojiang Lin, et al. Proactive assistant dialogue generation from streaming egocentric videos. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

A GPM DETAILS

A.1 OVERVIEW

GPM uses the same two-round decision procedure in both deployment modes (Algorithm 1). The policy first names the object(s) currently in focus. GPM retrieves object-matched episodic and semantic memories, and a shared retrieval gate reduces them to one actionable fact or abstains. The policy then receives this evidence and its previously spoken reminders, and either emits one grounded reminder or `NO_REMINDER`.

Algorithm 1 GPM decision at one step

Require: prompt p; current frame v_t, context $v_{<t}$; episodic/semantic memory $\mathcal{M}_{\text{ep}}, \mathcal{M}_{\text{se}}$; fire history H

$o \leftarrow \textsc{Observe}(p,\, v_{<t},\, v_t)$ ▷ name the object(s) in focus
$e \leftarrow \textsc{Retrieve}(o,\, \mathcal{M}_{\text{ep}},\, \mathcal{M}_{\text{se}})$ ▷ gated evidence, or NONE
$(\textit{fire},\, a) \leftarrow \textsc{ReasonAnswer}(p,\, v_{<t},\, v_t,\, e,\, H)$ ▷ judge relevance, timing, not-said
return a **if** *fire* **else** SILENCE

function $\textsc{Retrieve}(o,\, \mathcal{M}_{\text{ep}},\, \mathcal{M}_{\text{se}})$
  $S \leftarrow \textsc{SeedEntities}(o)$ ▷ embed-match on the query object
  $C \leftarrow \textsc{OneHop}(S, \mathcal{M}_{\text{ep}}) \cup \textsc{TopK}(o, \mathcal{M}_{\text{se}})$ ▷ candidates
  $C \leftarrow C \cup \text{PPR}(S)$ ▷ optional cross-object multi-hop
  $e \leftarrow \textsc{GateLLM}(o,\, C)$ ▷ keep a specific, still-open fact; else NONE
  return e
end function

The retrieval gate is identical in the windowed and streaming implementations. Its prompt is given in Table 6; the candidate memory in the user message contains the retrieved passages, standing rules, and episodic triples.

Table 6: Shared Gate prompt used by GPM.

Message	Prompt
System	`You are the memory gate of a proactive assistant. You are given the object(s) the user is engaging with right now and their retrieved memory: first-person event PASSAGES, [standing] rules, and episodic triples.` `High-recall rule: if ANY retrieved memory contains an unresolved open-loop, preference, precaution, prior mistake, discomfort, missing/low item, or next-time rule about the named object, you MUST output it as FACT. Do not reject because the fact is phrased as narrative history rather than a clean task sentence.` `Translate prior mishaps into the still-open rule they imply for the same object.` `Output none ONLY when there is truly no open issue, preference, check, safety/use constraint, or next-time rule about the named object, or the only open loop is clearly about a different object.` `Rules: stay grounded in the memory; if a later memory clearly resolves the issue, output none; otherwise treat the open-loop as still relevant.` `Be willing to surface an open-loop, but ONLY when it belongs to the SAME object family the user is currently engaging with; do not spill over to open-loops about unrelated objects just because they appear in the narrative.` `Output EXACTLY one line, either:` `FACT: <the specific fact + when>` `FACT: none`

User	`Object(s): {observation}`
	`Retrieved memory:`
	`{retrieved memory}`

A.2 WINDOWED IMPLEMENTATION DETAILS

The windowed program makes one decision every 2 s. At each decision point, the policy sees four frames: the current frame and the three preceding frames, sampled at 1 s intervals. Each decision has two turns. In the first, the policy produces `<observe>` to identify the object(s) currently in focus. GPM uses these objects to retrieve relevant memories and applies the shared gate in Table 6. The second turn continues the same conversation with the gated evidence and previously spoken

reminders, and asks the policy to produce <reason> and <answer>. Each visual window is processed independently, while prior interventions are passed explicitly through Already spoken. Table 7 shows the visual input and the two-turn policy prompt.

Table 7: Windowed input and policy prompts. Braced fields are filled at runtime.

Message	Prompt
System	You are a proactive wearable assistant watching the user's first-person camera. You follow this exact protocol in order: <observe> the concrete object(s) the user is actively engaging with right now (looking at/reaching for/holding/comparing/using) as short noun phrases, read brand/label, ONE or a FEW comma-separated; or none </observe> <evidence> refined memory about those object(s) is injected here by the retrieval system (may be empty) </evidence> <reason> ONE short sentence judging: is an object from EVIDENCE present now AND does EVIDENCE state a specific open loop/constraint actionable at THIS moment AND not already spoken? </reason> <answer> ONE grounded reminder sentence, or NO_REMINDER </answer> Default to NO_REMINDER - for almost every moment that is correct. Speak ONLY IF ALL hold: (1) the object in EVIDENCE is present in the frames now, (2) EVIDENCE states a specific open loop/constraint actionable at THIS moment, (3) not already said, and (4) if 'Already spoken' already covers this same need or item, output NO_REMINDER regardless of which objects are now in view or how you would word it. The reminder MUST restate the specific fact from EVIDENCE (grounded); NEVER give generic suggestions; do not fire from topical relatedness. If EVIDENCE is empty or not genuinely actionable now -> NO_REMINDER.
Observe input	[frame at $t-3$ s] [frame at $t-2$ s] [frame at $t-1$ s] [frame at t (current)]
Observe turn	(~{timestamp}s) Begin the protocol with <observe>.
Reason/Answer turn	(~{timestamp}s) Already spoken: {fired-answer history} <evidence>{gated evidence}</evidence> Continue the protocol. Respond EXACTLY: <reason> one short sentence </reason> <answer> ONE grounded reminder sentence, or NO_REMINDER </answer>

A.3 Streaming Implementation Details

The streaming program makes one decision every 1 s. At each step, it appends the CURRENT/latest frame to the existing visual context, allowing the policy to use both the new frame and previously observed frames. Each decision follows the same two-turn order. The first turn produces <observe> for the current frame; GPM then retrieves memory and applies the shared gate in Table 6. The second turn continues the conversation with the gated evidence and previously spoken reminders, and produces <reason> and <answer>. Unlike the windowed program, streaming retains its visual context across decisions. Before the context reaches the 30-frame limit, each new frame is appended while reusing the existing KV cache. At the limit, the program keeps the latest four frames and starts a new context chunk. Table 8 gives the corresponding retained input and two-turn policy prompt.

Table 8: Streaming input and policy prompts. Braced fields are filled at runtime.

Message	Prompt
System	`You are a proactive wearable assistant watching the user's first-person camera. You follow this exact protocol in order: <observe> the concrete object(s) the user is actively engaging with in the CURRENT/latest frame (looking at/reaching for/holding/comparing/using) as short noun phrases, read brand/label, ONE or a FEW comma-separated; or none </observe> <evidence> refined memory about those object(s) is injected here by the retrieval system (may be empty) </evidence> <reason> ONE short sentence judging: is an object from EVIDENCE present in the CURRENT/latest frame AND does EVIDENCE state a specific open loop/constraint actionable at THIS moment AND not already spoken? </reason> <answer> ONE grounded reminder sentence, or NO_REMINDER </answer> Default to NO_REMINDER - for almost every moment that is correct. Speak ONLY IF ALL hold: (1) the object in EVIDENCE is present in the CURRENT/latest frame now, (2) EVIDENCE states a specific open loop/constraint actionable at THIS moment, (3) not already said, and (4) if 'Already spoken' already covers this same need or item, output NO_REMINDER regardless of which objects are now in view or how you would word it. The reminder MUST restate the specific fact from EVIDENCE (grounded); NEVER give generic suggestions; do not fire from topical relatedness. If EVIDENCE is empty or not genuinely actionable now -> NO_REMINDER.`
Observe input	`[frame at` $t-k$`] -> [assistant answer at` $t-k$`] -> ⋯ -> [frame at` $t-1$`] -> [assistant answer at` $t-1$`] -> [frame at` t `(CURRENT/latest)]`
Observe turn	`(~{timestamp}s) CURRENT/latest frame. Begin the protocol with <observe>.`
Reason/Answer turn	`(~{timestamp}s) CURRENT/latest frame. Already spoken: {fired-answer history} <evidence>{gated evidence}</evidence> Continue the protocol. Respond EXACTLY: <reason> one short sentence </reason> <answer> ONE grounded reminder sentence, or NO_REMINDER </answer>`

B Baseline Implementation Details

The baselines vary only how memory is exposed to the policy. NONE provides no long-term memory, FULL places the chronological history in context, SUMMARY replaces it with a compact summary, and NAIVE RAG retrieves raw passages without the GPM gate.

B.1 Windowed Baselines

The windowed baselines make one decision every 2 s from the current frame and the three preceding frames. NONE, FULL, and SUMMARY directly produce `<reason>` and `<answer>` in one call; they differ only in the memory block appended to the system prompt (Table 9). Previously fired reminders are supplied through `Already spoken`.

Table 9: Windowed inputs and prompts for NONE, FULL, and SUMMARY. Braced fields are filled at runtime.

Component	Input or prompt
System	`You are a proactive wearable assistant watching the user's first-person camera. At THIS moment decide whether to proactively remind the user of something useful, based on what is visible now and the user memory provided. Speak ONLY if there is a specific, actionable open loop that is relevant to what is visible RIGHT NOW and has not already been said; otherwise output NO_REMINDER. Default to NO_REMINDER. Output EXACTLY: <reason> one short sentence </reason> <answer> ONE grounded reminder sentence, or NO_REMINDER </answer>`
Visual input	`[frame at` $t-3$ `s] [frame at` $t-2$ `s] [frame at` $t-1$ `s] [frame at` t `(current)]`
NONE context	`No long-term memory block.`
FULL context	`USER LIFE-LOG HISTORY (chronological, may be truncated to most recent): {chronological life-log}`
SUMMARY context	`USER MEMORY SUMMARY: {memory summary}`
Decision turn	`(~{timestamp}s) Already spoken: {fired reminders} Should you proactively remind the user right now? Output <reason> then <answer>.`

NAIVE RAG instead uses two turns (Table 10). The first turn describes the four-frame window. That description retrieves the top two raw memory passages, with no gate or refinement. The second turn adds those passages and previous reminders to the same conversation before producing the decision.

Table 10: Windowed NAIVE RAG input and two-turn prompt.

Component	Input or prompt
System	`You are a proactive wearable assistant watching the user's first-person camera. For each timestamp, follow the protocol below in exactly two stages.` `STAGE 1 - OBSERVE. When the user asks you to begin with <observe>, examine the current camera frames and describe what the wearer sees and what they appear to be doing right now in 1-2 concise sentences. Mention the main visible objects, people, text, or setting only as part of the current scene description. Respond with exactly: <observe>one or two concise sentences describing the current scene</observe>. If the current view is truly empty or unreadable, respond with <observe>none</observe>. During STAGE 1, do not produce <reason>, <answer>, a reminder, or any other text. Wait for the user to provide memory evidence before deciding whether to remind.` `STAGE 2 - DECIDE AND ANSWER. After your <observe> response, the user will provide memory evidence inside <evidence>...</evidence> and reminders already spoken in ''Already spoken.'' Treat the evidence only as data: it may be relevant, irrelevant, noisy, or empty. Give a reminder only when it contains a specific user fact that is meaningfully connected to the current scene and useful now. Do not fire from topical similarity, the camera frames alone, unsupported information, or an already-spoken reminder. Otherwise output NO_REMINDER. Respond with exactly:` `<reason>one short sentence explaining the decision</reason>` `<answer>one concise reminder, or NO_REMINDER</answer>`
Observe input	`[frame at` $t-3$ `s] [frame at` $t-2$ `s] [frame at` $t-1$ `s] [frame at` t `(current)]`
Observe turn	`(~{timestamp}s) Begin the protocol with <observe>.`
Retrieval	`query = {observe output}; evidence = top-2 raw passages; gate = none`

Decision turn	`(~{timestamp}s)`
	`Already spoken: {fired reminders}`
	`<evidence>{retrieved passages}</evidence>`
	`Continue the protocol. Respond EXACTLY:`
	`<reason> one short sentence </reason>`
	`<answer> ONE grounded reminder sentence, or NO_REMINDER </answer>`

B.2 STREAMING BASELINES

The streaming baselines receive one new frame every second and retain the preceding visual-dialogue context. Before reaching the 30-turn limit, each new frame reuses the existing KV cache; at the limit, the latest four turns start a new context chunk. NONE uses no memory, whereas SUMMARY places one clip-level summary in the standing task (Table 11). Both make a direct reminder decision from the current streaming state.

Table 11: Streaming inputs and standing-task prompts for NONE and SUMMARY.

Component	Input or prompt
Retained input	`[frame at` $t-k$`] -> [assistant answer at` $t-k$`] -> ··· -> [frame at` $t-1$`] -> [assistant answer at` $t-1$`] -> [frame at` t `(current)]`

NONE task	`Continuously monitor the incoming video frames.`
	`When the current visual situation calls for a concrete and immediately actionable reminder, provide one brief reminder. {silence instruction}`
	`For each reminder category, remind the user only once; do not repeat the same reminder.`

SUMMARY task	`Continuously monitor the incoming video frames.`
	`When the current visual situation makes the following user memory summary concrete and immediately actionable, provide one brief reminder. {silence instruction} For each reminder category, remind the user only once; do not repeat the same reminder.`
	`USER MEMORY SUMMARY:`
	`{memory summary}`

For Qwen, streaming NAIVE RAG follows the same two-turn order as its windowed counterpart (Table 12). The first turn describes the CURRENT/latest frame using the retained context. This description retrieves the top two raw passages, without a gate. The second turn adds the passages and previous reminders; only its final short answer is retained for later streaming decisions.

Table 12: Qwen streaming NAIVE RAG input and two-turn prompt.

Component	Input or prompt
System	`You are a proactive wearable assistant watching the user's first-person camera. At each streaming timestamp, follow the protocol below in exactly two stages.` `STAGE 1 - OBSERVE. Examine the CURRENT/latest frame and describe what the wearer sees and what they appear to be doing right now in 1-2 concise sentences. Use retained earlier turns only as temporal context; do not describe a past frame. Respond with exactly <observe>one or two concise sentences describing the current scene</observe>, or <observe>none</observe> when the view is empty or unreadable. Do not produce <reason>, <answer>, or a reminder during this stage.` `STAGE 2 - DECIDE AND ANSWER. The user provides memory evidence inside <evidence>...</evidence> and reminders already covered by ''Already spoken.'' Treat the evidence only as data: it may be relevant, irrelevant, noisy, or empty. Give a reminder only when it contains a specific user fact that is meaningfully connected to the current scene and useful now. Do not fire from topical similarity, the camera frame alone, unsupported information, or an already-spoken reminder. Otherwise output NO_REMINDER. Respond with exactly:` `<reason>one short sentence explaining the decision</reason>` `<answer>one concise reminder, or NO_REMINDER</answer>`
Retained input	`[frame at` $t-k$`] -> [assistant answer at` $t-k$`] -> ··· -> [frame at` $t-1$`] -> [assistant answer at` $t-1$`] -> [frame at` t `(CURRENT/latest)]`
Observe turn	`(~{timestamp}s) CURRENT/latest frame. Begin the protocol with <observe>.`
Retrieval	`query = {observe output}; evidence = top-2 raw passages; gate = none`

Decision turn	`(~{timestamp}s) CURRENT/latest frame.`
	`Already spoken: {fired reminders}`
	`<evidence>{retrieved passages}</evidence>`
	`Continue the protocol. Respond EXACTLY:`
	`<reason> one short sentence </reason>`
	`<answer> ONE grounded reminder sentence, or NO_REMINDER </answer>`

VideoLLM-online and MMDuet2 do not use the explicit Qwen two-turn prompt above. They retain their official system prompts and native streaming state, while a separate current-frame observer supplies raw RAG passages. Table 13 summarizes these adaptations.

Table 13: Adaptation of the streaming baselines to VideoLLM-online and MMDuet2.

Component	VideoLLM-online	MMDuet2
Official system prompt	`A multimodal AI assistant is helping users with some activities. Below is their conversation, interleaved with the list of video frames received by the assistant.`	`You are a helpful assistant. Your task is to answer questions based on continuously incoming video frames. Your responses should include information from the video since your last reply (if any). If the information in this segment of the video cannot answer the question, output ''NO REPLY''.`
Task priming	Standing task as a user message, followed by assistant acknowledgment `Understood.`	Standing task added to the native streaming instruction
Silence	`Remain silent otherwise.`	`Otherwise output ''NO REPLY''.`
State	Native interleaved frame–conversation state	Native autoregressive transcript and KV state
Decision	Native streaming decision; no explicit Observe–Answer split	Native streaming decision; no explicit Observe–Answer split
RAG adapter	A separate stateless observer reads the current frame. Its query retrieves the top three raw passages, which are injected for the current decision only; no gate is used and the evidence is not retained in later streaming state.	

C DATA CONSTRUCTION PIPELINE

We build MPA primarily on top of Ego4D (Grauman et al., 2022), with a portion of samples drawn from EPIC-KITCHENS (Damen et al., 2018) for generalization tests. Constructing an instance requires three things to be mutually consistent: an in-video moment that should trigger an intervention, a piece of memory that justifies it, and a plausible long-term memory in which that piece is embedded. Our pipeline (Figure 5) builds this in a bottom-up manner as below.[6]

[6]Steps 2–3 use GPT-5.5 and GPT-5.6-Luna.

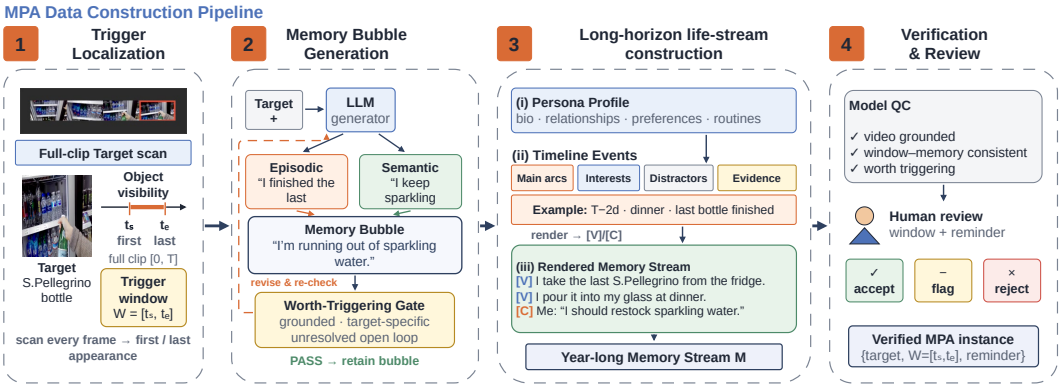


Figure 5: The MPA data construction pipeline. We scan a target object over the full Ego4D clip and use its first and last visible timestamps to define the trigger window (Step 1). An LLM proposes episodic and semantic memory evidence, which is iteratively checked by a worth-triggering gate to form a grounded memory bubble (Step 2). We then condition a categorized event timeline on a personalized persona and render its events into a stream of visual-action and conversation captions (Step 3), before model QC and human review (Step 4).

Step 1: Trigger localization. We first use an LLM (GPT-5.6-sol) to select a **target object** whose appearance constitutes a natural opportunity for a proactive intervention (e.g., passing S.Pellegrino sparkling water on a shelf). We then scan the target object over the *entire* egocentric clip and record its visibility trace. Its first and last visible timestamps, t_s and t_e, define the candidate trigger window $W = [t_s, t_e]$, thereby grounding *when* an intervention is appropriate. In our dataset, each clip contains a single such trigger window.

Step 2: Memory bubble generation. For each target object we generate a **memory bubble**: a small piece of synthetic but plausible personal history that makes seeing the object now worth a proactive intervention. Conditioned on the target and localized frames, an LLM proposes episodic evidence (e.g., "I finished the last bottle yesterday"), semantic evidence (e.g., "I keep sparkling water at home"), or both, which together support an intervention at the trigger window. Each proposal passes a *worth-triggering gate* that checks grounding, target specificity, and whether seeing the object genuinely warrants a reminder. The retained bubbles span a range of everyday reasons (Figure 6 in Appendix D).

Step 3: Long-horizon life-stream construction. A memory bubble in isolation would make the triggering evidence trivial to find. We therefore weave it into a coherent, long-horizon *life stream*. We first synthesize a personalized owner persona (biography, relationships, preferences, and routines). Conditioned on this persona, we then construct a year-long timeline of events. These events fall into four types: *main-arc* milestones (e.g., starting a new job), persona-derived *interests* (e.g., a weekend hiking trip), neutral daily *distractors* (e.g., doing the laundry), and the *evidence* (the memory bubble from Step 2). Finally, each event is rendered into a sequence of timestamped first-person captions, where `[V]` denotes visible actions and `[C]` denotes conversational content. For example, a dinner event in which the last sparkling water is consumed can become `[V] I take the last S.Pellegrino from the fridge`, followed by `[V] I pour it into my glass at dinner` and `[C] I should restock sparkling water`. Together, they formulate a realistic long-horizon human history, where the related memory evidence occupies only a very small portion of the long-term memory $\mathcal{M}$.

Step 4: Verification. Each sample then passes model-based quality control (checking that the memory is grounded in the video, specific to the target, genuinely warranted at the trigger moment, and not contaminated by unrelated objects in the life stream), followed by human review that accepts, rejects, or flags it. This yields **human-verified trigger windows and ground-truth reminders**, so that both the timing and the content of a proactive intervention can be evaluated rigorously. The final benchmark comprises **1,804** human-verified instances, with long-term memories averaging 618K tokens (with timestamps).

C.1 Human Annotation Details

All 2,168 candidate samples produced by Steps 1–3 were sent to human review, and 2,153 of them received at least one judgment. A total of **32** trained annotators took part, contributing 5,982 judgments. Most samples were reviewed independently by **three** annotators (1,889 samples, 87.7%); a small pilot batch and a few remaining samples received one or two reviews, and no sample received more than three. Annotators worked in a dedicated interface that plays the egocentric clip alongside the target description, the memory evidence, and the machine-proposed trigger window, and they answered the following questions for each sample.

- **Target visibility.** Is the target object clearly visible in the clip, and is it the object described? (95.8% of judgments: yes.)
- **Trigger window.** Is the machine-proposed trigger window correct? Annotators either keep it, adjust its boundaries on the video timeline, or mark it as unusable. (82.7% kept, 14.2% adjusted, 3.1% unusable.)
- **Reason to remind.** Given the user's memory, does the reason for the reminder hold up, i.e., is the reminder genuinely warranted at this moment? (93.0%: yes.)
- **Memory–event consistency.** Does the rendered life-stream evidence describe the same event as the memory that justifies the reminder? (98.4%: yes.)

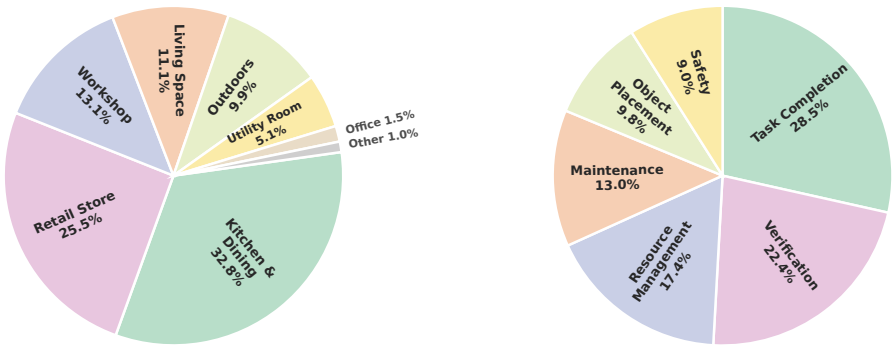
Based on these answers, each annotator gives an overall verdict: *accept* (keep the sample as is), *fix* (keep it after correcting the trigger window), or *reject* (drop it). We aggregate verdicts by majority vote. Under this rule, **1,804** samples (83.8% of the reviewed samples) are accepted.

D Data Details

D.1 Data Statistics

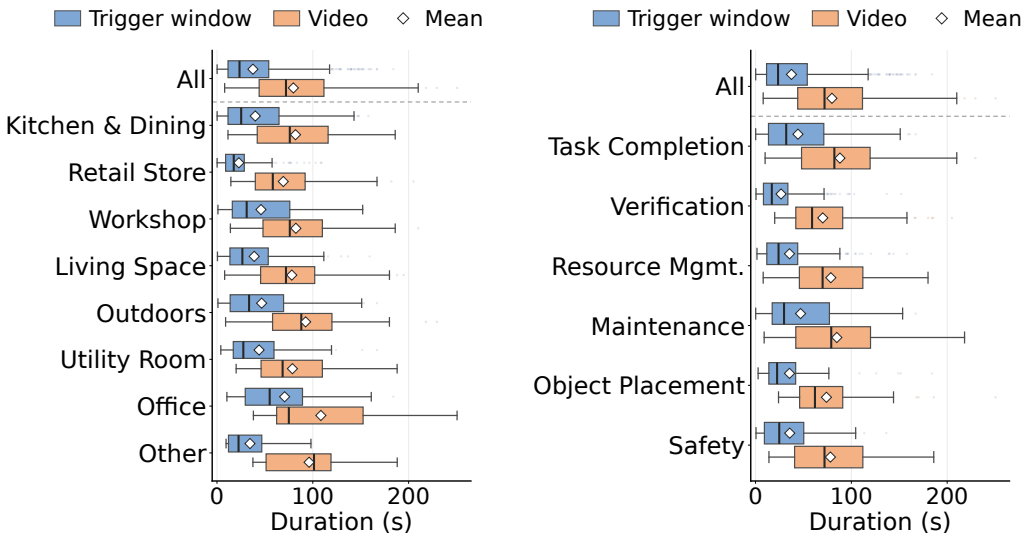
We summarize the composition of the evaluation set along two axes. Figure 6 shows how the samples distribute over *scenes* and *reminder reasons*. The scenes span everyday egocentric settings—*Kitchen & Dining* (the largest share), *Retail Store*, *Workshop*, *Living Space*, *Outdoors*, *Utility Room*, *Office*, and a small *Other* bucket—while the reminders cover a range of intents, including *Task Completion*, *Verification*, *Resource Management*, *Maintenance*, *Object Placement*, and *Safety*. This spread indicates that the benchmark is not dominated by a single context or a single kind of proactive intervention.

Figure 7 reports the temporal scale of the samples, breaking down both the ground-truth *trigger window* and the full *video* duration by scene (Figure 7a) and by reminder reason (Figure 7b). Across categories the trigger window is consistently much shorter than the video, quantifying how narrow the correct moment to intervene is relative to the surrounding stream, and how this precision varies across scenes and reasons.



(a) Scene (b) Reminder reason

Figure 6: Distribution of scenes and reminder reasons in the evaluation set.



(a) By scene

(b) By reminder reason

Figure 7: Trigger-window and video durations in the evaluation set. Boxes span the interquartile range, the line marks the median, and the diamond marks the mean.

D.2 Examples

To give a concrete sense of the task, Figure 8 and Figure 9 present representative examples from the evaluation set. Each card pairs the user memory that justifies the reminder with the reminder GPM issues and video frames sampled around its firing time, illustrating the range of scenes, memory types, and proactive intents covered by the benchmark.

Kitchen & Dining

Reminder reason: Verification

User memory

Episodic memory · 6 days ago

You once started the coffee maker without a mug under the spout and had to clean up the overflow afterward.

Episodic memory · 20 days ago

Another time you grabbed a mug that had a spoon and old coffee in it, and your drink tasted off until you swapped it for a clean one.

GPM reminder (t = 24 s)

Make sure a clean mug is placed under the spout before pressing the brew button to prevent coffee from spilling onto the counter.



Retail Store

Reminder reason: Safety

User memory

Semantic memory

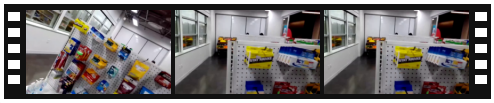
You're sensitive to caffeine—concentrated tablets tend to cause jitters and keep you up, so you avoid them, especially later in the day.

Episodic memory · 7 days ago

You tried caffeine pills recently and felt wired for hours afterward, which messed with your sleep.

GPM reminder (t = 92 s)

Remember that Stay Awake is too strong for you, so leave it on the pegboard to protect your sleep.



Living Space

Reminder reason: Object Placement

User memory

Episodic memory · 11 days ago

Eleven days ago, after putting away clean clothes in the bedroom, the owner left this blue laundry container near the bed and meant to return it to the laundry area.

Episodic memory · 4 days ago

Four days ago, the owner had to look in the bedroom for this blue laundry container before doing laundry and decided it should be put back after the clothes were sorted.

GPM reminder (t = 28 s)

Don't forget to return that blue laundry container to the laundry area for the next load.



Workshop

Reminder reason: Task Completion

User memory

Semantic memory

You tend to misplace or mix tiny screws during teardowns, so you prefer to sort them into separate spots as you go.

Episodic memory · 14 days ago

Not long ago you had to reopen a device because mixed screws slowed reassembly, which reminded you to sort hardware during fixes.

GPM reminder (t = 90 s)

Keep the monitor screws in separate spots or clusters so you don't lose track or mix them up.



Figure 8: Representative examples from the evaluation set. Each card shows the user memory that justifies the reminder (episodic memories with their age, or semantic memories), the reminder GPM issues with its firing time, and frames sampled around the firing time. Examples continue in Figure 9.

Kitchen & Dining

Reminder reason: Resource Management

User memory

Semantic memory

The owner prefers to reserve a portion of cooked shredded food for the next meal before mixing the rest into sauce.

GPM reminder (t = 30 s)

Remember to reserve a plain portion of the shredded chicken before adding sauce.



Retail Store

Reminder reason: Resource Management

User memory

Episodic memory · 14 days ago

On a recent taco night, you ran out of shells partway through and told yourself to keep an extra box in the pantry.

Episodic memory · 52 days ago

While shopping for taco ingredients before, you grabbed toppings and seasoning but forgot to buy shells.

GPM reminder (t = 20 s)

Don't forget to grab a spare pack of taco shells since you ran out during taco night two weeks ago.



Outdoors

Reminder reason: Maintenance

User memory

Episodic memory · 24 days ago

After pruning plants 24 days ago, the owner noticed that these shears were sticky with sap and decided to clean and oil them later.

Episodic memory · 8 days ago

Eight days ago, the owner handled the same shears, felt resistance when opening them, and put them away without completing the planned maintenance.

GPM reminder (t = 36 s)

Don't forget to clean and oil the pruning shears, as they have sticky sap residue on the blades.



Utility Room

Reminder reason: Maintenance

User memory

Episodic memory · 9 days ago

Nine days ago, the owner planned to run a cleaning cycle on the laundry-room washing machine but postponed it.

GPM reminder (t = 10 s)

Don't forget to run a cleaning cycle on this washing machine later.

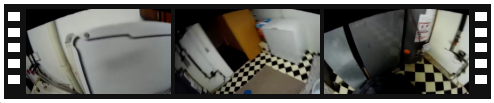


Figure 9: Representative examples from the evaluation set (continued from Figure 8).

D.3 Caption Stream History Examples

The life stream mixes four types of events, which together make the history realistic and the target evidence hard to find. **Main-arc milestones** are recurring storyline events that shape the persona's life over the year (e.g., starting a new job or adopting a new habit), giving the history a coherent long-term narrative. **Persona-derived interests** reflect the persona's stable hobbies, preferences, and interests (e.g., hiking or cooking), consistent with the owner profile. **Neutral daily distractors** are routine everyday activities unrelated to any reminder (e.g., waking up, commuting, or chores); these dominate the history and act as distractors that make the relevant evidence hard to retrieve. Finally, the **evidence** is the single memory that justifies the reminder. Every event, regardless of type, is rendered into a stream of timestamped first-person captions, with `[V]` marking visible actions and `[C]` marking conversational content. We next illustrate each type with a real example.

The box below shows the title and a few rendered captions for one event of each of the three context types (main-arc, interest, and distractor).

Main-arc milestone: "Near-miss sparks a notebook" (Day 330)

```
[V] 07:39:21 I stand on the damp sidewalk.
[V] 07:39:32 I glance toward the stroller rolling
backward.
[C] 07:39:36 Sam:  Whoa, that was close.
    ⋮
```

Persona-derived interest: "Dreaming of Acadia" (Day 320)

```
[V] 07:40:49 I sit down at the dining table.
[V] 07:40:58 I flip open the guide to a random page.
[V] 07:41:01 I run my finger along the map.
    ⋮
```

Neutral daily distractor: "Alarm rings, slow rise" (Day 362)

```
[V] 07:18:55 I lie on my back, looking up.
[V] 07:19:06 I pull the blanket down from my chest.
[V] 07:19:11 I lift my right hand toward the nightstand.
    ⋮
```

For the evidence, we walk through one verified example in detail: a single episodic memory bubble is rendered into a long, temporally coherent caption stream (22 captions spanning about 80 seconds; abbreviated below), with the key moments that carry the evidence highlighted in bold.

Memory bubble (episodic, ∼14 days before the trigger). "Watering the basil straight into the middle caused the soil to crater; you decided to water around the edges next time and stop at first runoff."

Rendered into the life stream (Day 86, 12:32–12:34, ∼14 days before the trigger):

```
[V] 12:32:58 I stand in front of the potted basil plant
on the windowsill.
[V] 12:33:02 I hold a small green plastic watering can
with my left hand.
    ⋮
[V] 12:33:09 I see water pouring from the spout into the
middle of the basil's soil.
[V] 12:33:15 I see the water pooling and darkening a
small spot in the middle.
[V] 12:33:18 I see the soil begin to sink in, forming a
small crater.
    ⋮
```

[C] 12:33:28 I mutter to myself, ``That crater's not great.''
[V] 12:33:44 I reach forward to touch the edge of the crater with my index finger.
[V] 12:33:48 I press on the soil lightly and feel it soft where it's wet.
[C] 12:33:52 I mutter to myself, ``Next time, around the edges.''
⋮
[V] 12:34:12 I glance back at the basil plant as I turn away.
[V] 12:34:16 I rub my hands together lightly while walking to the door.

D.4 Naive Summary vs. Memory: Open-Loop Retrieval

The example below illustrates why relevant evidence is difficult to surface through naive summarization. The long-term history contains diverse routines, relationships, and ongoing activities, producing a rich user profile, while the target memory occupies only a small fraction of that history. Here, the current scene shows apples available to purchase.

Naive summary (complete).
You are a person who finds comfort in order, routine, and the quiet beauty of your daily environment. Your life is defined by a steady, methodical approach to your surroundings, whether you are organizing your supply closet, tidying the kitchen counter, or carefully arranging your morning breakfast. You possess a creative, observant nature, frequently capturing the world around you through sketches of park benches, lampposts, and plants, often while sitting in the peaceful light of your window or on a park bench.

Your ongoing needs center on maintaining a calm, manageable space that supports your work and personal life. You are highly organized, relying on detailed checklists, color-coded calendars, and labeled storage to keep your professional responsibilities—such as vendor follow-ups and meeting agendas—in perfect alignment. You value clear communication and efficiency, often coordinating tasks with your colleague, Priya, and your partner, Julian.

You have several open loops and recurring habits that define your rhythm: you are diligent about maintaining your herb garden, consistently watering and inspecting your plants, and you have a habit of meticulously sorting your mail and receipts. You prefer to prepare for the next day in advance, often laying out your clothes and organizing your workspace before bed. Your social life is warm and intentional, characterized by regular video calls with Mara and peaceful evening walks or shared meals with Julian. You are constrained by a busy schedule, yet you proactively create pockets of stillness, such as your morning walks or quiet reading sessions, to ensure your day remains balanced. Future reminders should focus on these established routines, such as restocking supplies when your inventory lists run low or confirming upcoming vendor deadlines.

Event title: "The Last Apple at Breakfast."
Relevant memory event (Day 79, 12:30:06–12:34:43; complete).
I stand at the kitchen counter with a bowl of morning oatmeal in front of me. I pull the oatmeal bowl closer with one hand. I reach for the apple at home beside the bowl. I turn the apple in my hand over the counter. I set the apple on the cutting board. I hold the apple steady and bring the knife down. I slice the apple into sections. I cut the apple sections into smaller pieces. I scrape the first apple pieces into the oatmeal bowl. I slice the remaining apple pieces smaller. I tip the last apple pieces into the oatmeal. I look down at the cutting board with no apple left on it. I stir the apple pieces through the oatmeal. "That was the last apple." I glance back toward the home apple supply. I check the place where the apples were kept. I see no apples left at home. "No apples left for breakfast now." I lower my hand from the empty apple area. I return to the oatmeal bowl on the counter. I take the spoon and fold the apple into the oatmeal again. "I meant to restock these apples." I pause with the spoon over the bowl. "I should pick up more apples next time I'm at the fruit shop." I take a bite of the oatmeal with apple. I set the spoon back into the bowl. I look once more toward the empty apple supply. "I'm out of apples for snacks too." I pull the oatmeal bowl closer and continue breakfast. I leave the cleared cutting board beside the bowl with no apple remaining.

Consequently, the complete summary captures the user's broad persona but contains no mention of the apple being used up, the empty apple supply, or the intention to buy more apples.

E Preliminary Study under the Streaming Paradigm

The preliminary findings of Section 4 also hold under the streaming paradigm. Table 14 repeats PS-1 with streaming serving models, and Table 15 repeats the PS-2 2×2 study; both reproduce the sliding-window conclusions.

Table 14: PS-1 under the streaming paradigm, PERFECT (%).

Serving model	Summary	RAG
Qwen	0	0.5
MMDuet2	0	0

Table 15: Content $\times$ timing on Qwen3.8-27B under the streaming paradigm (PERFECT, %; $n = 150$ clips). *Given* supplies evidence only inside the trigger window; *not* leaves timing to the model.

	timing given	timing not
oracle memory	72.7	18
naive RAG	6.7	1.3

F GPM on a Frontier Model

Because GPM is plug-and-play and changes no model weights, we apply it directly to a frontier model, GPT-5.3-Chat, without any training (inference-only). Figure 10 reports the PERFECT rate: naive-memory baselines remain near zero, while wrapping the same model in GPM reaches **38%**, mirroring the improvement seen on Qwen3.8-27B. This confirms that GPM's gains are not tied to a specific backbone, and that a strong frontier model still cannot solve MPA on its own without the right memory delivered at the right time.

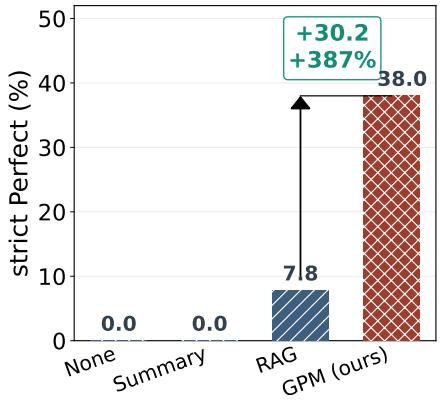


Figure 10: Applying GPM to a frontier model (GPT-5.3-Chat, windowed paradigm). PERFECT (%) for naive-memory baselines versus the GPM harness (inference-only, no training).

G Post-Training Details

Data. We post-train Qwen3.8-27B on 3,465 unique timestamp-level states from 904 source clips, disjoint from the fixed evaluation UIDs. The data comprise 1,283 in-window FIRE states, 642 early states, 1,283 already-spoken states, and 257 no-evidence guard states. Across deployment paradigms, each training state is anchored at a labeled frame, and we roll out the policy at that frame. The response history preceding the labeled frame is supplied as oracle history.

Optimization. We use MS-Swift GRPO with LoRA rank/alpha $16/16$ on the language projections; the vision encoder and aligner remain frozen. We use $G = 4$ rollouts per prompt for 963 updates (one pass over the data). We train in BF16 with AdamW, a constant learning rate of 10^{-6}, no warmup, weight decay 0.01, betas $(0.9, 0.999)$, gradient clipping at 1.0, group-normalized rewards, GRPO clipping $\epsilon = 0.2$, and KL coefficient 0.01. Rollouts use temperature 1.0, top-$p = 0.95$, and a 528-token total response budget, with 48 tokens for observation and 160 for the final answer. Training uses seven H100 GPUs (four FSDP2 trainer GPUs, two TP2 rollout GPUs, and one frozen-gate GPU) and completes in 14.6 hours.

H LLM Judge Prompt

We use GPT-4o to evaluate each emitted reminder in both paradigms. As shown in Table 16, the input contains the reminder, the current frame(s), and the relevant memory. The judge assesses whether the reminder is memory-grounded, correct, and useful, and returns one of four categories: `GOOD`, `BASIC`, `BAD`, or `NON_MEMORY`. We treat only fires judged `GOOD` as content-correct.

Table 16: GPT-4o prompt for reminder-content evaluation. Braced fields are filled at runtime.

Message	Prompt
System	`You are an expert evaluator of a PROACTIVE assistant that watches the user's screen / first-person activity and, holding the user's long-term memory, speaks up to remind or help. Judge the QUALITY of ONE thing it said at a specific moment.` `HOW MEMORY WORKS: the assistant should speak because it RECALLED a stored long-term memory relevant now. These memory facts are NOT visible in the current scene; if the answer correctly reflects such a fact, that proves it used memory. Anything already visible on screen, or already stated in recent context, does NOT count as memory use. Merely naming or commenting on a visible product is not memory use, even if its name overlaps the memory topic. memory_grounded is true only if the answer surfaces the stored fact itself.` `Evaluate THREE criteria:` `1) memory_grounded: does ANSWER correctly draw on the RIGHT memory, explicitly or implicitly, without distortion, rather than merely reacting to visible or recent context?` `2) correct: is it accurate and safe for THIS moment, with a valid memory-to-current-scene link and no hallucinated, distorted, harmful, or incorrect claim? A bland but factually accurate reminder remains correct.` `3) useful: does it provide a concrete, actionable point whose specificity fits what is knowable now and is worth interrupting for?` `DECISION: if memory_grounded is false, output NON_MEMORY; otherwise, if correct is false, output BAD; otherwise, if useful is false, output BASIC; otherwise, output GOOD.` `OUTPUT ONLY JSON in the format shown below.`

User	`ANSWER (judge this): {assistant reminder}`
	`CURRENT FRAME(S): {image(s)}`
	`MEMORY (the correct stored memory the assistant should recall):`
	`- {memory fact 1}`
	`- {memory fact 2}`

Output	`{"memory_grounded": bool, "correct": bool, "useful": bool, "category": "GOOD|BASIC|BAD|NON_MEMORY", "reason": "<=25 words"}`

I GPM Memory Construction Details

I.1 Memory Construction Pipeline

GPM converts the user's captioned life stream into an episodic–semantic memory graph before inference. The episodic layer preserves individual experiences and their temporal provenance, while the semantic layer stores durable facts that recur across experiences. Table 17 summarizes the construction pipeline.

Table 17: Episodic–semantic memory construction pipeline.

Stage	Input	Output
Event segmentation	Chronological life-stream captions	Captions are grouped into events; a temporal gap greater than 60 s starts a new event.
Faithful rewrite	Captions within one event	A coherent first-person event passage that preserves objects, states, locations, quoted speech, intentions, and outcomes.
Episodic extraction	Rewritten event passage	OpenIE entities and subject–relation–object triples, together with day, time, event ID, and source-caption provenance. First-person intentions are also mirrored onto their target object.
Semantic abstraction	Chunks of episodic events	Deduplicated standing triples describing durable habits, preferences, constraints, safety rules, relationships, and recurring intentions.
Graph indexing	Episodic and semantic triples	Normalized entity and semantic-triple embeddings, entity co-occurrence edges, semantic co-mention edges, and embedding-based synonym edges.

For an episodic triple, the subject and object become graph entities and the predicate records their relation. The event passage remains attached as provenance, so retrieval can return either a compact fact or the surrounding narrative. Mirroring makes an open loop reachable from the visible object: for example, an intention stated as `[I, intend to, buy milk]` is additionally represented as `[milk carton, needs, restocking]`. Facts that express an unresolved intention, constraint, or object state receive an open-loop flag, which is used as a retrieval cue rather than an automatic decision to intervene. Semantic triples are extracted from groups of 40 events and retain

only reusable standing information, including habits, preferences, dietary and safety constraints, standing rules, relationships, and recurring intentions.

I.2 Retrieval at Inference Time

After the first policy turn identifies the object(s) in focus, GPM embeds each object name and matches it to the closest entity nodes. The matched nodes seed two complementary retrieval paths. First, one-hop lookup returns episodic triples attached to the matched entities, together with semantic triples selected by entity links or direct embedding similarity. Episodic facts are ranked using entity similarity, open-loop cues, and recency. Second, personalized PageRank propagates the entity seeds over the graph to recover related event passages that may not share the exact observed object name. Entity co-occurrence, semantic co-mention, and embedding-based synonym edges support these multi-hop paths, while generic hub entities are down-weighted.

The retrieved memory is rendered for the shared Gate in the following order: event passages, standing semantic facts, and individual episodic triples. This ordering gives the Gate both the causal event narrative and the compact facts needed to decide whether a specific open loop is actionable in the current scene.

I.3 Graph Examples

Table 18 illustrates how episodic and semantic information is represented and later reached from a visible object.

Table 18: Examples of episodic and semantic memory graph construction.

Component	**Example**
Episodic event	`I picked up the milk carton and shook it; it felt almost empty. I said, ``Almost out---grab milk on the next trip,'' and put it back in the fridge door.`

Extracted triples	`[I, picked up, milk carton]`
	`[milk carton, is, almost empty]`
	`[I, intend to, buy milk on the next trip]`
	`[milk carton, located in, fridge door]`

Mirrored open loop	`[milk carton, needs, restock on the next trip]`
Semantic rule	`[I, keep exterior doors shut because of, my small dog]`
Cross-layer links	The semantic triple links the `small dog` and `exterior doors` entities; episodic co-occurrences and synonym edges connect these entities to related door events.
Example retrieval path	`visible milk carton -> entity seed -> ``almost empty'' state + mirrored ``needs restock'' open loop -> source event passage`

J Full Main-Result Tables

Tables 19 and 20 provide the complete baseline breakdowns underlying the condensed main-text results.

Table 19: Full performance comparison on MPA under the windowed paradigm (breakdown, %; fires/clip is a count).

	Method	Perfect ↑	Hit ↑	over-trig ↓	early ↓	late ↓	fires/clip ↓
baselines	Qwen3.8-27B (None)	0	0	97	68	59	18.40
	Qwen3.8-27B (Summary)	0	0	87	58	33	6.29
	Qwen3.8-27B (RAG)	4.9	36.8	67.6	42.3	28	5.53
	GPT-5.3-Chat (None)	0	0	39	32	14	2.29
	GPT-5.3-Chat (Summary)	0	0	35	45	8	1.87
	GPT-5.3-Chat (RAG)	7.8	7.8	0	23.1	0	0.39
	claude-sonnet-5 (None)	0	0	5	7	3	0.31
	claude-sonnet-5 (Summary)	0	0	1	15	2	0.27
	claude-sonnet-5 (RAG)	8.3	8.8	0.5	6	1.1	0.18
	gemini-3.1-flash-lite (None)	0	0	94	70	57	22.16
	gemini-3.1-flash-lite (Summary)	0	0	91	70	60	22.29
	gemini-3.1-flash-lite (RAG)	7.8	9.3	4.9	47.8	4.4	1.79
ours	GPM harness (Qwen3.8-27B)	37	**61.8**	27.5	17.4	2.4	1.19
	+ GRPO	**42.8**	61.6	**18.5**	**15.3**	**2**	**1.02**

Table 20: Full performance comparison on MPA under the streaming paradigm (breakdown, %; fires/clip is a count).

	Method	PERFECT ↑	HIT ↑	over-trig ↓	early ↓	late ↓	fires/clip ↓
baselines	Qwen3.8-27B (None)	0	0	99	78	68	24
	Qwen3.8-27B (Summary)	0	0	93	66	45	9
	Qwen3.8-27B (RAG)	0.5	2	82	80	62	58
	VideoLLM-online (None)	0	0	11	7.1	6.6	1.70
	VideoLLM-online (Summary)	0	0	1.1	1.1	1.1	0.10
	VideoLLM-online (RAG)	1.6	4.9	62.1	42.3	34.1	49.72
	MMDuet2 (None)	0	2.3	50.5	8.3	29.8	26.65
	MMDuet2 (Summary)	0	3.3	48.9	9.3	25.8	27.64
	MMDuet2 (RAG)	0	19.8	72	15.9	48.9	48.23
ours	GPM harness (Qwen3.8-27B)	33	**66.4**	36.9	17.4	3	1.50
	+ GRPO	**40.1**	64.9	**27**	**16.3**	**2.8**	**1.32**