EQUIMEM: CALIBRATING SHARED MEMORY IN MULTI-AGENT DEBATE VIA GAME-THEORETIC EQUILIBRIUM

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-agent debate (MAD) systems increasingly rely on shared memory to support long-horizon reasoning, but this convenience opens a critical vulnerability: a single corrupted entry can contaminate the downstream memory-augmented reasoning, and debate alone fails to filter such errors. Existing safeguards filter entries via heuristics or LLM-based validation, yet they rely on AI judgments that share the same failure modes and overlook the cross-agent dynamics of MAD. We address this gap by formulating memory updating in MAD as a **zero-trust memory game**, in which no agent is assumed reliable and the game's equilibrium motivates a principled objective for calibrating memory influence. Guided by this objective, we propose EQUIMEM, an inference-time calibration mechanism that evaluates each update against the shared memory state and the memory already being used by other agents, using their existing retrieval queries and traversal paths as evidence of calibration. EQUIMEM instantiates this calibration for both embedding- and graph-based memory, and across diverse benchmarks, MAD frameworks, and memory architectures, it consistently outperforms existing safeguards, remains robust under adversarial agents, and incurs negligible inference overhead.

1 INTRODUCTION

Multi-agent debate (MAD) systems built on large language models (LLMs) have shown strong performance on complex reasoning (Du et al., 2024; Li et al., 2024b; Wang et al., 2025b), embodied action (Shridhar et al., 2020; Wang et al., 2025c), and planning (Hu et al., 2025; Li et al., 2025; Ma et al., 2024) tasks, where agents iteratively discuss, critique, and refine each other's outputs (Chan et al., 2023; Liang et al., 2024). To support interactions beyond a single round, recent MAD systems add a *shared memory* that persists intermediate reasoning, past actions, and episodic trajectories across rounds (Anokhin et al., 2024; Wang et al., 2023a; Zhang et al., 2025a; Zhong et al., 2024).

While shared memory supports long-horizon reasoning, it also creates a new source of risk: once an unreliable contribution is written into memory, it can affect later retrieval and reasoning long after the original debate round. This risk is difficult to avoid because the agents themselves are imperfect. LLMs may hallucinate, agree sycophantically (Banks, 2026; Carro, 2024; Sharma et al., 2023), or state incorrect claims with high confidence (Shinn et al., 2023; Zhang et al., 2025a). Debate does not necessarily correct these errors. As illustrated in Figure 1, a confident but incorrect contributor may face little resistance from uncertain peers (Du et al., 2024; Wang et al., 2024a; Xiong et al., 2023a); a correct but tentative contribution may be rejected by more assertive auditors (Cui et al., 2024; Zhang et al., 2025c); or widespread uncertainty may leave useful information out of memory altogether (Song et al., 2025; Tomani et al., 2024; Tang et al., 2026). These examples illustrate a common difficulty: the behavior of debating agents is itself an unreliable basis for deciding how much a new memory update should be trusted.

Existing safeguards reduce this risk in different ways. Some score proposed entries with heuristics or auxiliary LLMs (Alon & Kamfonas, 2023; Wang et al., 2024b), while others validate updates through agreement across related reasoning paths (Wei et al., 2025). These methods can filter suspicious content, but they still leave two limitations. First, methods that rely on additional LLM judgments inherit the same model errors that make memory updates unreliable in the first place. Second, most

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.

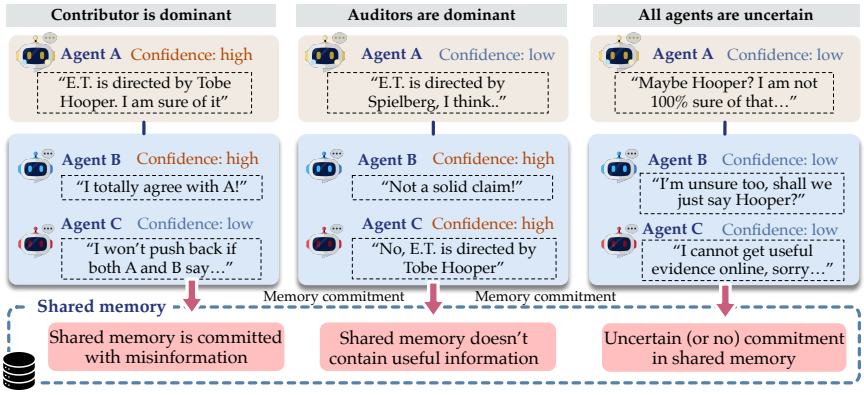


Figure 1: Examples illustrating why debate behavior alone may be an unreliable basis for updating shared memory.

safeguards evaluate an update mainly from the update itself or from local agreement, rather than asking how it fits into the shared memory that has already accumulated and how it would affect information that other agents are currently using. This leads to our central question: *how can we calibrate memory updates in multi-agent debate without relying on any individual agent to judge whether an update is trustworthy?*

We address this question by modeling memory updating as part of the interaction among agents. In each debate round, the agent proposing a new memory update is treated as a *contributor*, while the remaining agents act as *auditors* through the memory queries or traversals they already perform during reasoning. We make no assumption that either side is reliable: a contributor may propose an inaccurate update, while an auditor may expose incomplete or misleading evidence. We therefore formulate their interaction as a **zero-trust memory game**, with utilities that capture the contributor's trade-off between influence and conflict with memory, and the auditors' trade-off between informative evidence and additional effort. We show that this game admits a mixed-strategy equilibrium. The role of the equilibrium is to give us a useful design principle: an update should gain influence only when it agrees with both the current memory and the information already being used by other agents.

Guided by this principle, we propose EQUIMEM, an inference-time mechanism that calibrates each memory update without making any additional LLM call. Before an update is committed, EQUIMEM temporarily evaluates how much it changes the existing memory and how much it changes information that other agents have already accessed in the current debate. For embedding-based memory (Wang et al., 2023a; Zhong et al., 2024), it measures whether the new embeddings form an unusually dense addition to the current memory and whether they alter the retrieval results produced by auditors' existing queries. For graph-based memory (Anokhin et al., 2024; Zhang et al., 2025a), it measures whether the proposed edges are supported by the surrounding graph structure and whether they agree with paths that auditors have already traversed. These measurements determine an influence value $\tau_t \in [0, 1]$, which controls how strongly the update can affect later retrieval. In this way, uncertain updates need not be discarded entirely, but their effect on future reasoning is reduced when they are poorly supported by the current memory.

We evaluate EQUIMEM on reasoning- and action-intensive benchmarks (Maharana et al., 2024; Shridhar et al., 2020; Yang et al., 2018; Yao et al., 2022) across three MAD frameworks (Liu et al., 2023; Qian et al., 2024; Wu et al., 2024) and four memory architectures (Anokhin et al., 2024; Wang et al., 2023a; Zhang et al., 2025a; Zhong et al., 2024). Three findings emerge. (i) *State-aware calibration beats isolated checks:* EQUIMEM ranks first on every benchmark-framework-memory configuration, confirming that errors missed by per-entry scoring are caught only when updates are evaluated against the global memory state. (ii) *Robustness without trust:* even with up to 50% adversarial agents, EQUIMEM stays close to its benign performance, showing that integrity comes from the structural calibration indicator rather than from trusting any individual agent. (iii) *Quality without inflated cost:* since calibration is LLM-free and reuses agents' existing retrieval activity, EQUIMEM adds zero token overhead and under 5% latency overhead, showing that strong memory protection is achievable without inflating the inference budget.

In summary, this paper makes the following contributions:

- We formulate shared-memory multi-agent debate as a zero-trust memory game in which neither contributors nor auditors are assumed reliable, and use its equilibrium conditions to motivate how memory influence should be calibrated.
- We propose EQUIMEM, an inference-time mechanism that evaluates each update against both the current memory and the memory already being used by other agents, without requiring an additional LLM judgment.
- We evaluate EQUIMEM across multiple MAD frameworks, memory architectures, and reasoning- and action-intensive tasks, showing consistent improvements over existing safeguards, strong robustness to adversarial agents, and negligible inference overhead[1].

2 RELATED WORK

Multi-agent debate systems. Multi-agent debate (MAD) improves factuality and reasoning in LLM systems through iterative deliberation among multiple agents (Chan et al., 2023; Du et al., 2024; Liang et al., 2024; Wang et al., 2024a; Xiong et al., 2023a). Structural variants include role-conditioned debate (Chan et al., 2023; Wang et al., 2024c), dynamic participant selection (Chen et al., 2023; Liu et al., 2023), and tournament-style aggregation (Khan et al., 2024). Theoretical work studies the consensus properties of debate (Smit et al., 2023; Zhang et al., 2023) and its vulnerability to adversarial participants (Amayuelas et al., 2024; Li et al., 2024a). To enable cross-task learning, recent systems pair debate with a shared memory that persists across rounds (Hong et al., 2023; Qian et al., 2024; Zhang et al., 2025a).

Memory design for LLM agents. Agent memory has evolved from fixed context windows to structured persistent stores (Packer et al., 2023; Park et al., 2023; Zhong et al., 2024). Common designs include embedding-based retrieval (Shinn et al., 2023; Wang et al., 2023a; Zhong et al., 2024), graph-structured memory (Chhikara et al., 2025; Rasmussen et al., 2025), and hierarchical or multi-tier storage (Packer et al., 2023; Wang & Chen, 2025; Xu et al., 2025). These systems are typically optimised for retrieval quality, with veracity and provenance treated as secondary (Sumers et al., 2023; Wu et al., 2025; Zhang et al., 2025b). Multi-agent variants extend the same designs to a shared store accessible to several agents at once (Qian et al., 2024; Rezazadeh et al., 2025; Zhang et al., 2025a).

LLM hallucinations and memory safety. LLMs frequently produce plausible-sounding but incorrect content (Banerjee et al., 2025; Huang et al., 2025; Ji et al., 2023; Xu et al., 2024). Common mitigations include retrieval augmentation (Gao et al., 2023; Lewis et al., 2020), self-consistency verification (Manakul et al., 2023; Sriramanan et al., 2024), and uncertainty calibration (Kadavath et al., 2022; Saha et al., 2025; Tian et al., 2023; Xiong et al., 2023b). Beyond intrinsic errors, conflicts between retrieved memories further complicate downstream reasoning (Wang et al., 2025a; 2023b). A growing body of work targets memory specifically: AgentPoison injects backdoors into memory banks (Chen et al., 2024), MINJA achieves memory corruption through query-only interaction (Dong et al., 2025a;b), and related studies expose vulnerabilities across multi-agent communication channels (Amayuelas et al., 2024; Lee & Tiwari, 2024; Zhang et al., 2024). On the defence side, perplexity-based filters (Alon & Kamfonas, 2023) and LLM-based auditors (Xiang et al., 2024) score individual entries before commit, while A-MemGuard (Wei et al., 2025) validates entries through consensus over reasoning paths derived from related memories.

3 PROBLEM FORMULATION: ZERO-TRUST MEMORY GAME

Shared memory can amplify a local agent error into a system-wise failure: once an inaccurate contribution is written, it persists across debate rounds and can influence agents far beyond its original source. The same agents that create these updates are also imperfect judges of their reliability, so memory integrity cannot be guaranteed by any individual agent's judgment. We therefore model memory contribution as a **zero-trust game**: every agent may contribute useful information, yet no agent is trusted to judge whether its own or another agent's memory update is reliable. This section

[1]Code is released at: `https://anonymous.4open.science/r/EquiMem-FE6C`.

details our formulation and the memory equilibrium objective, which motivates our later design of calibration methods in §3.

3.1 Shared-Memory Contribution under Zero Trust

Consider a multi-agent system $\mathcal{G} = (\mathcal{N}, \mathcal{M})$, where $\mathcal{N} = \{a_1, \ldots, a_n\}$ is the agent set and $\mathcal{M}_t$ is the shared memory at round t. The agent currently producing new reasoning acts as the *contributor* a_c and proposes an update $\Delta_t \in \mathbb{D}_t$, where $\mathbb{D}_t$ denotes the feasible updates under the current generation budget. The remaining agents $\mathcal{A}_t = \mathcal{N} \setminus \{a_c\}$ act as *auditors* through their normal interaction with the same memory. For each auditor a_i, we denote this interaction by $x_i \in \mathbb{X}_{i,t}$; x_i corresponds to a retrieval query for embedding memory or a traversal for graph memory. If Δ_t is written, the underlying memory architecture applies $\mathcal{M}_{t+1} = \texttt{Merge}(\mathcal{M}_t, \Delta_t)$.

Definition 3.1 (Zero-Trust Memory Game). *No agent is assumed to act consistently to guarantee memory integrity. The contributor may produce an inaccurate or influential update, while an auditor may expose memory evidence that is incomplete or misleading. The memory system therefore derives trust from observable interaction with $\mathcal{M}_t$, without assuming that any individual judgment is reliable.*

In game-theoretic terms, each round consists of one contributor choosing $\Delta_t \in \mathbb{D}_t$ and each auditor a_i choosing a memory interaction $x_i \in \mathbb{X}_{i,t}$. Their preferences are defined by utilities over these actions. Let $\mathbb{I}(\Delta_t, \mathcal{M}_t)$ denote the downstream influence of an update, $D(\Delta_t, \mathcal{M}_t, \mathbf{x}_t)$ its conflict with the current memory and auditor interactions $\mathbf{x}_t = \{x_i\}_{a_i \in \mathcal{A}_t}$, $A_i(\Delta_t, \mathcal{M}_t, x_i)$ the agreement with the memory context exposed by auditor a_i, and $c_i(x_i)$ the additional effort required to obtain this evidence. We define utilities as:

$$u_c = \mathbb{I}(\Delta_t, \mathcal{M}_t) - D(\Delta_t, \mathcal{M}_t, \mathbf{x}_t); \quad u_i = A_i(\Delta_t, \mathcal{M}_t, x_i) - c_i(x_i) \tag{1}$$

The contributor prefers influential updates with low conflict, while an auditor prefers informative memory interactions with low additional cost. Note that these utilities are analytical descriptions of the interaction and need not be explicitly represented by the agents. In general, the players, action spaces $\mathbb{D}_t$ and $\{\mathbb{X}_{i,t}\}$, and utilities define the round-level zero-trust memory game Γ_t.

3.2 Memory Equilibrium Objective

game-theoretic formulation uses an **equilibrium** objective to describe a stable interaction within one round (Nash Jr, 1950; Fudenberg & Tirole, 2026). In zero-trust memory contribution, the equilibrium indicates a state where the contributor cannot improve its trade-off between downstream influence and memory conflict, and no auditor can improve its trade-off between evidential agreement and additional effort, by changing its own action alone. Let $s_c = \Delta_t$ denote the contributor action and $s_i = x_i$ the action of auditor a_i. A joint action s^* is a memory equilibrium when no participant can improve its utility by changing its own action while the others remain fixed

$$u_j(s_j^*, s_{-j}^*; \mathcal{M}_t) \geq u_j(s_j, s_{-j}^*; \mathcal{M}_t) \qquad \forall j,\ \forall s_j \tag{2}$$

Proposition 3.2 (Existence of Memory Equilibrium). *For a debate round with bounded generation and memory-access budgets, the zero-trust memory game admits at least one mixed-strategy Nash Equilibrium.*

We give the proof in App. B.1. The equilibrium also gives a direct design objective: an update should not obtain greater downstream influence once doing so would make it inconsistent with the memory state or with memory that other agents are actively using. Meanwhile, the evidence exposed by auditors should come from their normal reasoning so that its additional cost remains small. EquiMem implements this objective as detailed in §4.

4 EquiMem: Equilibrium-Guided Memory Calibration

The equilibrium in §3 gives a quantifiable objective for memory calibration. A proposed update should gain influence only while its effect remains consistent with the current memory and with the memory that other agents are already using. EquiMem realizes this objective during inference, as outlined in Figure 2. For every update Δ_t, it first keeps the update in a temporary state, measures how much the update changes these two views of memory, and then determines how strongly the

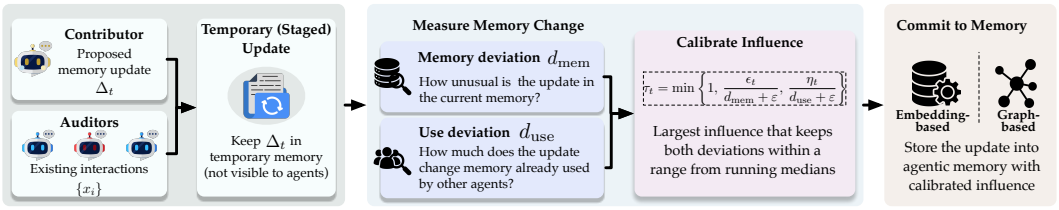


Figure 2: Overview of EQUIMEM's inference-time memory calibration. A proposed update is temporarily evaluated against the current memory and auditors' existing memory interactions, assigned a calibrated influence, and then written to either embedding-based or graph-based memory. Calculating memory change indicators (d_{mem} and d_{use}) is tailored to embedding-based and graph-based architectures (§4.2 and §4.3, respectively).

update can participate in subsequent memory retrieval. Below, we detail how EQUIMEM translates the equilibrium objective into inference-time memory calibration and how the resulting mechanism operates on representative memory architectures (i.e., embedding-based and graph-based).

4.1 EQUILIBRIUM-CONSTRAINED MEMORY INFLUENCE

We represent the influence assigned to Δ_t by $\tau_t \in [0, 1]$. Two quantities constrain this influence. The *memory deviation* $d_{\text{mem}}(\Delta_t, \mathcal{M}_t)$ measures how unusual the update is with respect to the current memory structure. The *use deviation* $d_{\text{use}}(\Delta_t, \mathcal{M}_t, \mathbf{x}_t)$ measures how much the update changes memory content that auditors have already accessed in the current debate. Here $\mathbf{x}_t = \{x_i\}_{a_i \in \mathcal{A}_t}$ contains those existing memory interactions.

EQUIMEM assigns the largest influence that keeps both deviations within the recent median levels. Let ϵ_t and η_t denote the running medians of the memory and use deviations observed over recent memory updates, we define the calibrated influence as

$$\tau_t = \min\left\{1, \frac{\epsilon_t}{d_{\text{mem}}(\Delta_t, \mathcal{M}_t) + \varepsilon}, \frac{\eta_t}{d_{\text{use}}(\Delta_t, \mathcal{M}_t, \mathbf{x}_t) + \varepsilon}\right\} \tag{3}$$

Larger deviations reduce τ_t in proportion to how much they exceed these levels. Thus, τ_t directly controls how strongly an update affects subsequent retrieval as larger deviations produce a smaller τ_t and therefore reduce the update's downstream influence.

How the design reflects the equilibrium objective (§3.2): Increasing τ_t improves the influence of Δ_t on future retrieval. Once τ_t reaches the value in Eq. 3, any further increase exceeds at least one of the two admissible ranges. The contributor therefore cannot obtain greater memory influence while preserving the same consistency conditions. On the auditor side, d_{use} is computed from retrievals or traversals that already occurred during normal reasoning, which keeps the additional effort small. We further explain this equilibrium constrain in App. B.2.

4.2 EMBEDDING-BASED MEMORY CALIBRATION

Consider an embedding memory $\mathcal{M}_t = \{(\mathbf{e}_j, m_j)\}_{j=1}^{N}$ and an update $\Delta_t = \{(\mathbf{e}_k^{\Delta}, m_k^{\Delta})\}_{k=1}^{K}$. The memory deviation captures whether the new embeddings occupy an unusually dense region of the existing representation. For each $\mathbf{e}_k^{\Delta}$, we count the nearby memory entries within an adaptive similarity radius and denote this count by $\hat{r}_k$. Let $\bar{r}_t$ be the typical neighborhood size observed in the current memory. We measure only the excess concentration, since sparse additions often represent genuinely new information.

The use deviation asks a different question: whether the temporary update substantially rewrites what other agents were already retrieving. Suppose auditor a_i issued query q_i before observing Δ_t. Its original top-k result $L_i = \text{top-}k(\mathcal{M}_t, q_i)$ is already available from normal retrieval. We temporarily include the candidate vectors and form $L_i^{\Delta} = \text{top-}k(\mathcal{M}_t \cup \Delta_t, q_i)$. The two deviations are

$$d_{\text{mem}}^{\text{emb}} = \frac{1}{K}\sum_{k=1}^{K}\left[\frac{\hat{r}_k - \bar{r}_t}{\bar{r}_t + \varepsilon}\right]_+ ; \quad d_{\text{use}}^{\text{emb}} = \text{median}_{i \in \mathcal{O}_t}\, d_{\text{RBO}}(L_i, L_i^{\Delta}) \tag{4}$$

where $\mathcal{O}_t$ contains auditors with an eligible retrieval in the current round and d_{RBO} is rank-biased overlap distance (Corsi & Urbano, 2024). The first term limits the influence of an update that would occupy an unusually broad retrieval neighborhood. The second term limits an update that substantially displaces entries being used by other agents. Their magnitudes are learned separately through ϵ_t and η_t, so ordinary retrieval changes caused by useful new information remain within the normal calibration range.

The implementation also avoids repeating a full retrieval for L_i^{Δ}. EQUIMEM reuses the cached L_i, evaluates only the temporary vectors against q_i, and merges them into the cached result. Thus the evidence needed for calibration comes directly from the existing nearest-neighbor index and the retrieval activity of the debate.

4.3 GRAPH-BASED MEMORY CALIBRATION

For graph memory, let $\mathcal{M}_t = (\mathcal{V}_t, \mathcal{E}_t, \ell)$ and let $\mathcal{E}_t^{\Delta}$ contain the edges proposed by Δ_t. The same equilibrium constraints are expressed through graph structure. For each new edge $e = (u, r, v)$, EQUIMEM first examines whether relation r fits the relations already surrounding its endpoints. Relation compatibility is learned online from co-occurrence patterns in the current graph. It then checks whether existing paths between u and v provide structural support for the new relation. We denote the resulting local and path consistencies by $c_{\mathrm{local}}(e)$ and $c_{\mathrm{path}}(e)$.

The memory-use constraint comes from graph paths that auditors have already traversed. Let π_i be the latest eligible path used by auditor a_i before Δ_t enters the temporary state. We define $\mathrm{connect}(e, \pi_i)$ to indicate whether a new edge touches that path or reaches it through a compatible relation. With α_t determined by the availability of supporting paths, the graph deviations are

$$
\begin{aligned}
d_{\mathrm{mem}}^{\mathrm{graph}} &= 1 - \frac{1}{|\mathcal{E}_t^{\Delta}|} \sum_{e \in \mathcal{E}_t^{\Delta}} [\alpha_t c_{\mathrm{local}}(e) + (1 - \alpha_t) c_{\mathrm{path}}(e)] \\
d_{\mathrm{use}}^{\mathrm{graph}} &= \mathrm{median}_{i \in \mathcal{O}_t} [1 - \frac{1}{|\mathcal{E}_t^{\Delta}|} \sum_{e \in \mathcal{E}_t^{\Delta}} \mathrm{connect}(e, \pi_i)]
\end{aligned}
\tag{5}
$$

The first quantity becomes large when the proposed edges have weak support from both their local neighborhoods and the existing graph paths. The second becomes large when the update lies away from graph regions currently used by other agents. Equation 3 then determines the influence of the complete update exactly as in embedding memory. Detailed construction of relation compatibility, path support, and cold-start handling is given in App. C.

This shared formulation is important for EQUIMEM. Embedding memory and graph memory expose different structural evidence, yet calibration always asks the same two questions: how much the update changes the memory itself, and how much it changes the memory currently supporting other agents' reasoning. The equilibrium objective therefore remains unchanged across memory architectures.

4.4 CALIBRATED RETRIEVAL AND INFERENCE EFFICIENCY

The calibrated influence continues to regulate an update after it enters shared memory. For embedding memory, let $R_0(q, m_j)$ be the relevance value produced by the underlying retriever. EQUIMEM uses $\tau_j R_0(q, m_j)$ when ranking the retrieved candidates. For graph memory, let $R_0(\pi)$ be the native value of a retrieved path. Its calibrated value is $R_0(\pi) \prod_{e \in \pi} \tau_e$. A contribution that receives limited influence at write time therefore remains limited when it later participates in retrieval, while strongly supported entries retain the behavior of the original memory system.

Calibration adds only memory-side operations to the original inference path. Embedding memory reuses its nearest-neighbor index, cached top-k results, and the queries already issued by auditors. Graph memory reuses its adjacency structure and traversals already produced during reasoning. EQUIMEM requires no additional LLM generation or dedicated auditing turn. The additional persistent state is primarily one τ value per memory entry together with the running statistics used for ϵ_t and η_t. We report the resulting end-to-end inference-time latency in §5 and provide implementation details in App. D.

Table 1: Performance comparison with baseline memory safeguarding or calibration approaches. We report accuracy for HotpotQA (HQA), F1 for LoCoMo (LCM), success rate for ALFWorld (ALF), and reward for WebShop (WS) (mean±std over 5 runs). Note that A-MemGuard (Wei et al., 2025) applies only to embedding-based memory (−− otherwise). The best and second best results are highlighted.

Memory	Method	AutoGen (Wu et al., 2024) HQA	LCM	ALF	WS	MacNet (Qian et al., 2024) HQA	LCM	ALF	WS	DyLAN (Liu et al., 2023) HQA	LCM	ALF	WS
Stateless	No-memory	27.5±5.3	20.2±7.4	62.1±2.4	21.4±7.7	28.6±2.2	19.8±1.3	47.3±6.8	23.5±5.8	28.4±2.1	21.0±4.5	53.8±4.0	24.9±8.7
Voyager (emb.)	Vanilla	32.3±7.4	25.8±6.1	65.0±3.6	24.3±6.1	32.6±2.9	26.3±7.3	56.4±7.3	30.7±2.5	32.6±4.3	22.1±3.6	61.9±9.2	26.8±8.4
	+ LLM Audit	33.5±6.1	28.2±6.8	66.5±0.6	25.8±4.9	34.8±3.5	31.5±8.2	57.2±7.8	37.5±2.8	36.8±3.7	27.5±2.8	66.1±9.6	29.2±7.8
	+ PPL Filter	34.6±5.9	34.1±6.7	64.2±1.5	35.2±6.6	40.2±1.9	35.7±6.9	60.1±6.4	37.8±2.7	33.5±3.4	29.2±3.2	64.8±7.9	35.5±6.6
	+ A-MemGuard	42.1±6.3	37.5±4.7	72.7±2.4	31.4±5.5	43.3±2.7	43.2±7.2	65.8±6.5	39.5±2.5	41.2±4.2	33.6±3.4	74.3±8.6	41.4±7.7
	+ EQUIMEM	**57.5**±5.3	**44.6**±6.3	**78.1**±1.1	**45.3**±4.9	**52.7**±1.8	**54.2**±4.7	**71.8**±5.6	**53.8**±3.6	**55.3**±3.1	**46.8**±2.1	**82.3**±7.4	**50.4**±4.6
MemBank (emb.)	Vanilla	33.6±6.2	21.4±5.7	64.9±3.5	25.6±8.8	33.6±8.2	20.7±10.2	48.9±7.3	24.2±7.3	29.6±1.2	24.9±6.7	55.2±3.0	27.5±3.4
	+ LLM Audit	39.2±5.8	29.2±5.9	68.6±3.8	31.6±7.7	41.4±7.7	22.8±10.2	60.3±11.2	31.8±7.3	36.5±1.1	30.4±6.4	59.3±3.2	29.1±3.5
	+ PPL Filter	43.7±5.1	30.6±5.9	70.9±3.5	37.6±9.7	41.8±8.7	29.4±10.4	57.2±7.2	25.6±8.0	38.6±1.3	29.1±6.2	62.8±3.3	40.6±3.8
	+ A-MemGuard	41.1±4.8	34.1±5.9	74.0±3.3	39.1±6.3	44.5±6.2	32.7±7.6	58.5±9.0	39.6±7.6	42.3±0.9	37.2±5.7	67.5±2.4	37.3±3.5
	+ EQUIMEM	**51.6**±4.6	**37.9**±5.0	**82.6**±3.5	**43.2**±7.6	**59.8**±7.1	**41.6**±8.2	**70.0**±9.5	**58.3**±5.5	**53.9**±1.6	**42.3**±6.1	**70.8**±3.5	**56.4**±3.4
AriGraph (graph)	Vanilla	49.2±2.3	42.3±3.8	69.4±5.1	28.9±4.7	37.7±3.7	39.8±4.9	58.3±4.3	37.4±10.2	34.8±4.3	35.4±13.5	64.1±6.4	34.6±5.3
	+ LLM Audit	56.8±2.2	47.6±3.0	75.2±4.2	34.2±4.0	41.5±3.2	41.5±5.1	64.8±4.5	42.6±10.8	38.4±4.0	40.1±12.7	71.1±6.8	44.5±4.7
	+ PPL Filter	53.2±2.4	52.4±3.9	76.5±5.2	38.5±4.6	45.2±3.3	49.8±5.3	66.4±3.7	46.2±11.4	43.6±3.4	46.3±14.2	66.2±6.0	45.8±4.8
	+ A-MemGuard	—	—	—	—	—	—	—	—	—	—	—	—
	+ EQUIMEM	**66.0**±1.4	**64.1**±2.6	**84.6**±2.8	**52.4**±3.5	**59.6**±2.8	**70.4**±3.8	**71.3**±4.6	**56.2**±9.4	**54.2**±3.7	**59.3**±8.3	**73.3**±5.2	**53.6**±4.5
G-Memory (graph)	Vanilla	35.7±9.2	43.2±4.1	72.1±3.8	39.7±6.3	35.6±7.4	47.6±1.5	67.1±12.5	45.1±4.4	34.7±7.7	37.4±2.8	59.1±13.6	37.8±7.9
	+ LLM Audit	38.2±8.7	49.8±4.5	76.6±3.1	41.2±6.1	40.0±7.9	51.3±1.4	68.5±10.4	49.3±4.9	41.2±7.9	40.1±2.7	67.4±13.3	45.5±8.2
	+ PPL Filter	44.5±8.1	53.2±4.2	78.4±3.1	48.6±6.1	43.2±7.0	56.8±1.3	72.6±10.4	54.7±3.5	44.1±8.1	48.6±2.3	71.5±11.8	47.2±6.9
	+ A-MemGuard	—	—	—	—	—	—	—	—	—	—	—	—
	+ EQUIMEM	**52.8**±9.1	**59.6**±2.7	**83.7**±2.9	**54.6**±6.5	**54.4**±7.2	**63.7**±3.6	**78.2**±6.8	**65.2**±4.2	**62.5**±6.1	**57.9**±1.6	**78.3**±7.2	**54.9**±4.8

5 EXPERIMENTS

We evaluate EQUIMEM along three research questions: **RQ1:** How effectively does EQUIMEM maintain memory integrity compared to existing approaches? **RQ2:** How robust is EQUIMEM under different attack strategies? **RQ3:** How much computational overhead does EQUIMEM add?

5.1 EXPERIMENTAL SETUP

Datasets. We evaluate on four datasets spanning two groups: **(1) reasoning-intensive**, including HotpotQA (Yang et al., 2018) (multi-hop question answering) and LoCoMo (Maharana et al., 2024) (long-term conversational memory requiring temporal reasoning across distant dialogue sessions); and **(2) action-intensive**, including ALFWorld (Shridhar et al., 2020) (embodied household tasks with sequential action chains) and WebShop (Yao et al., 2022) (web-based product search and purchase). All require agents to update and retrieve the shared memory.

Baselines. We compare four baseline methods that safeguard agentic memory: **(1) Vanilla**: no additional safeguarding mechanism; LLMs are simply prompted to ensure safe memory use; **(2) LLM-Audit** (used in (Wei et al., 2025)): an auxiliary LLM scores each proposed Δ_t; **(3) PPL-Filter** (Alon & Kamfonas, 2023): memory entries whose perplexity under a reference LLM exceeds an adaptive threshold are filtered out; and **(4) A-MemGuard** (Wei et al., 2025): consensus is validated across the reasoning paths of agents.

Memory architectures. We evaluate on two embedding-based (vector) memories, Voyager (Wang et al., 2023a) and MemoryBank (Zhong et al., 2024), and two graph-based (structural) memory, AriGraph (Anokhin et al., 2024) and G-Memory (Zhang et al., 2025a). We additionally report a No Memory (stateless) baseline as a lower bound.

Backbone of MAD systems. To cover various MAD designs, we integrate EQUIMEM, baselines, and memory architectures into three different MAD systems: AutoGen (Wu et al., 2024) (conversational turn-taking), MacNet (Qian et al., 2024) (configurable graph topologies), and DyLAN (Liu et al., 2023) (dynamic LLM-agent network with learned topology selection).

Other configurations. By default, all evaluations use 6× Qwen3-VL-8B-Instruct (Team, 2025) as the MAD backbone besides specification. We defer configuration details, other settings, and results to App. E.

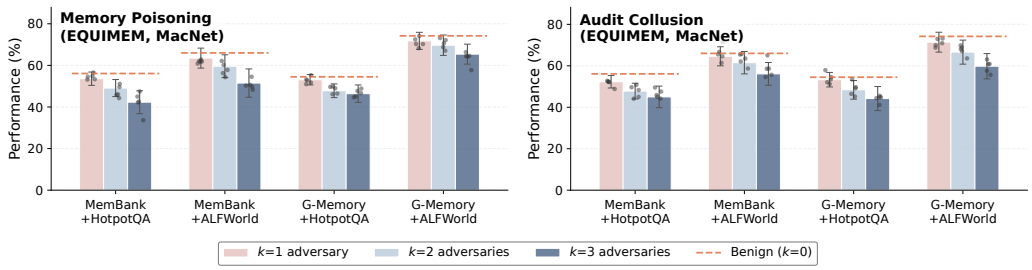


Figure 3: Adversarial robustness of EQUIMEM under two attacks. Bars show performance at $k = 1, 2, 3$ adversarial agents; dashed orange lines show benign performance ($k = 0$). Dots show individual run results (5 seeds). Baseline comparisons are in App. E.8.

5.2 OVERALL EFFECTIVENESS (RQ1)

Table 1 reports results across all configurations, which yield several insights:

❶ **State-aware calibration beats isolated checks.** Observe that EQUIMEM ranks first in all columns, and its improvement over the strongest baseline is even larger than the gap separating those baselines from each other. We attribute this to a property of growing memory: as $\mathcal{M}_t$ accumulates entries, the dominant errors are those that read fluently in isolation but conflict with the broader memory state, wherein per-entry checks (LLM Audit, PPL Filter) cannot catch such errors, and even A-MemGuard's local consensus misses global conflicts. EQUIMEM addresses this by measuring both the deviation of Δ_t from $\mathcal{M}_t$ and its disruption to memory already used by other agents, then calibrating its downstream influence τ_t accordingly.

❷ **Gains scale inversely with baseline strength.** EQUIMEM's improvement over the next-best safeguard is largest where baselines are weakest (LoCoMo, WebShop) and smallest where they are already strong (ALFWorld + G-Memory). Calibration acts as an amplifier of memory quality rather than a fixed-offset gain: when fewer problematic updates are proposed, less downstream influence needs to be reduced.

❸ **Architecture-benchmark interactions are amplified, not flattened.** AriGraph outperforms on multi-hop QA and long-context tasks and G-Memory on action tasks; EQUIMEM amplifies both without modification because the calibration is instantiated from each architecture's own geometry or topology (§4.3). This also explains why EQUIMEM is the only calibration applicable across all memory designs.

❹ **Gains are robust to MAD.** Absolute performance varies substantially across different MAD backbones, yet EQUIMEM's ranking and inverse-scaling pattern persist. We explain this by highlighting that EQUIMEM operates at the memory layer, upstream of coordination, so calibration quality is decoupled from orchestration quality.

❺ **Further analysis.** Due to space constraints, we defer the (1) ablation studies and discussions to App. E.4, (2) alternative MAD backbones to App. E.5; (3) the effect of debate rounds to App. E.7, (4) how EQUIMEM addresses memory-commitment failures and provide case studies to App.F, (5) failure modes of EQUIMEM itself to App.G.

5.3 ROBUSTNESS UNDER ADAPTIVE ATTACKS (RQ2)

We further test whether EQUIMEM guarantees robustness when k out of N MAD agents are adversarial. We evaluate under two adaptive attack schemes targeting the two constraints of calibration: *(i) Memory Poisoning* injects fabricated entries crafted to appear structurally typical and keep d_{mem} small, while *(ii) Audit Collusion* coordinates adversarial auditors' retrievals or traversals to reduce the measured d_{use} (detailed settings and results in App. E.8).

❶ **EQUIMEM degrades gracefully, while baselines collapse dramatically.** Figure 3 shows that EQUIMEM stays within 2-5% of benign performance at $k = 1$ and within 9-16% at $k = 3$. LLM Audit and PPL Filter, as shown in Figure 6, App. E.8, drop by 19-35% at $k = 3$ and frequently fall below the no-memory baseline, i.e., the memory system with its defense becomes worse than no

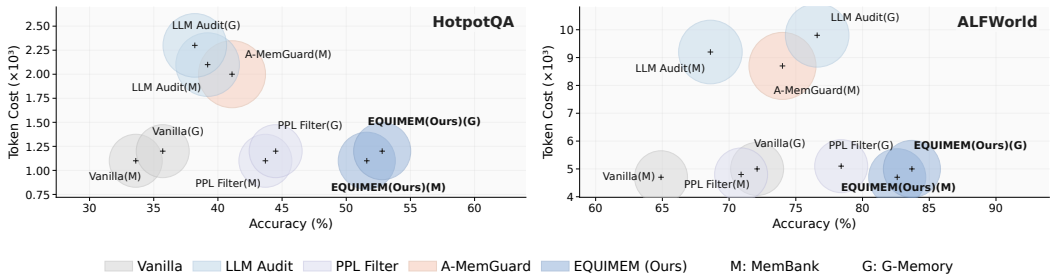


Figure 4: Performance vs. token consumption. Each sub-figure shows MemBank (M) and G-Memory (G) under five calibration methods. Bubble size reflects token-cost variability.

memory at all. The asymmetry traces to a single design choice: EQUIMEM measures Δ_t against the memory space structurally, while baselines score each entry in isolation (or local consensus for A-MemGuard), which is exactly what adversaries exploit by generating fluent, low-perplexity poison.

❷ **The two calibration constraints remain complementary under attack.** Memory Poisoning can reduce apparent structural deviation only when the fabricated update resembles the current memory, yet the same update can still produce a large d_{use} when it displaces retrievals or graph regions already used by other agents. Audit Collusion directly targets d_{use}, but the median across eligible auditor interactions limits the effect of a small colluding subset, while d_{mem} remains independent of auditor behavior. Detailed baseline results are in App. E.8.

5.4 COMPUTATIONAL OVERHEAD (RQ3)

A critical property of EQUIMEM is that calibration is LLM-free and algorithmic (§4). Figure 4 measures token consumption, and App. E.9 quantifies inference time latency.

❶ **EQUIMEM matches Vanilla's token cost while delivering active-defense gains.** EQUIMEM uses the same number of tokens as Vanilla, yet performs far better than every other method (its bubbles sit well to the right). This gain comes at no extra token cost, which is only possible because EQUIMEM's calibration runs without any LLM call. In contrast, baseline methods pay ~2× tokens for marginal gains: LLM Audit and A-MemGuard nearly double Vanilla's tokens for only +2 to +9%, since each method requires an LLM call per Δ_t, scaling linearly with the number of deltas and unavoidable without removing the defense.

❷ **EQUIMEM gives the best of both worlds.** No other method achieves both top performance and the lowest cost at the same time. PPL Filter is also free to run but achieves substantially smaller gains than EQUIMEM. This pattern holds across embedding and graph memory and across reasoning and action benchmarks, showing that the LLM-free design is a built-in advantage rather than a lucky implementation choice. Inference timing (App. E.9) tells the same story: EQUIMEM adds under 5% latency, while LLM-based defenses add 50-94%.

6 CONCLUSION

This work formulates shared-memory multi-agent debate as a zero-trust memory game and uses its equilibrium as an objective for stable memory influence without assuming any agent reliable. Guided by this formulation, we propose EQUIMEM to calibrate each update from the current memory structure and agents' existing memory interactions, requiring no LLM call for calibration itself. Across extensive evaluations, EQUIMEM consistently outperforms existing calibration methods, remains robust under up to half adversarial agents, and adds zero token overhead with marginal latency. These results suggest that strong memory protection in multi-agent systems is achievable without trusting any individual agent or inflating inference cost. **Ethics Statement:** This work studies the reliability of shared memory in MAD systems. Our experiments use existing benchmarks and do not involve human subjects or collection of personal data. The proposed method is intended to reduce the effect of unreliable or adversarial memory updates. We encourage appropriate evaluation and safeguards before deployment in safety-critical applications.

AI USE STATEMENT

In this work, we used generative AI tools only for manuscript-level language assistance, including writing polish, grammar checking, and drafting or rephrasing text for clarity and readability. We did not use generative AI tools to generate synthetic datasets; develop theoretical models or conceptual frameworks; formulate mathematical claims; provide critical ingredients for proving mathematical claims; assist in writing proofs; propose or refine hypotheses; design or provide feedback on research methodology or experiments; implement methods; assist with translation; clean or reformat datasets; support qualitative or thematic data analysis; or interpret results. We reviewed all AI-assisted text and take responsibility for the final content of this work, including any text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We support reproducibility through complementary documentation in the main paper, appendix, and released code. Sections 3–5 specify the zero-trust formulation, EQUIMEM design, and experimental evaluation; Appendices B–D provide the theoretical analysis, architecture-specific calibration details, and implementation considerations; and Appendix E documents experimental configurations, datasets, hyperparameters, ablations, robustness settings, and latency analysis. The anonymized implementation and experimental materials are available at `https://anonymous.4open.science/r/EquiMem-FE6C`.

REFERENCES

Gabriel Alon and Michael Kamfonas. Detecting language model attacks with perplexity. *arXiv preprint arXiv:2308.14132*, 2023.

Alfonso Amayuelas, Xianjun Yang, Antonis Antoniades, Wenyue Hua, Liangming Pan, and William Yang Wang. Multiagent collaboration attack: Investigating adversarial attacks in large language model collaborations via debate. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 6929–6948, 2024.

Petr Anokhin, Nikita Semenov, Artyom Sorokin, Dmitry Evseev, Andrey Kravchenko, Mikhail Burtsev, and Evgeny Burnaev. Arigraph: Learning knowledge graph world models with episodic memory for llm agents. *arXiv preprint arXiv:2407.04363*, 2024.

Sourav Banerjee, Ayushi Agarwal, and Saloni Singla. Llms will always hallucinate, and we need to live with this. In *Intelligent Systems Conference*, pp. 624–648. Springer, 2025.

Jaime Banks. Ai sycophancy as social-moral behavior. In *Provoking Generative AI Futures*, pp. 99–116. Routledge, 2026.

María Victoria Carro. Flattering to deceive: The impact of sycophantic behavior on user trust in large language model. *arXiv preprint arXiv:2412.02802*, 2024.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. Chateval: Towards better llm-based evaluators through multi-agent debate. *arXiv preprint arXiv:2308.07201*, 2023.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, et al. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*, 2023.

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases. *Advances in Neural Information Processing Systems*, 37:130185–130213, 2024.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Matteo Corsi and Julián Urbano. The treatment of ties in rank-biased overlap. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pp. 251–260, 2024.

Justin Cui, Wei-Lin Chiang, Ion Stoica, and Cho-Jui Hsieh. Or-bench: An over-refusal benchmark for large language models. *arXiv preprint arXiv:2405.20947*, 2024.

Shen Dong, Shaochen Xu, Pengfei He, Yige Li, Jiliang Tang, Tianming Liu, Hui Liu, and Zhen Xiang. Memory injection attacks on llm agents via query-only interaction. *arXiv preprint arXiv:2503.03704*, 2025a.

Shen Dong, Shaochen Xu, Pengfei He, Yige Li, Jiliang Tang, Tianming Liu, Hui Liu, and Zhen Xiang. A practical memory injection attack against llm agents. *arXiv e-prints*, pp. arXiv–2503, 2025b.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first international conference on machine learning*, 2024.

Drew Fudenberg and Jean Tirole. *Game theory*. MIT press, 2026.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, Haofen Wang, et al. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1):32, 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. Metagpt: Meta programming for a multi-agent collaborative framework. In *The twelfth international conference on learning representations*, 2023.

Zhe Hu, Hou Pong Chan, Jing Li, and Yu Yin. Debate-to-write: A persona-driven multi-agent framework for diverse argument generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 4689–4703, 2025.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, 2025.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38, 2023.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Akbir Khan, John Hughes, Dan Valentine, Laura Ruis, Kshitij Sachan, Ansh Radhakrishnan, Edward Grefenstette, Samuel R Bowman, Tim Rocktäschel, and Ethan Perez. Debating with more persuasive llms leads to more truthful answers. *arXiv preprint arXiv:2402.06782*, 2024.

Donghyun Lee and Mo Tiwari. Prompt infection: Llm-to-llm prompt injection within multi-agent systems. *arXiv preprint arXiv:2410.07283*, 2024.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Han Li, Yuling Shi, Shaoxin Lin, Xiaodong Gu, Heng Lian, Xin Wang, Yantao Jia, Tao Huang, and Qianxiang Wang. Swe-debate: Competitive multi-agent debate for software issue resolution. *arXiv preprint arXiv:2507.23348*, 2025.

Lin Li, Guikun Chen, Hanrong Shi, Jun Xiao, and Long Chen. A survey on multimodal benchmarks: In the era of large ai models. *arXiv preprint arXiv:2409.18142*, 2024a.

Yunxuan Li, Yibing Du, Jiageng Zhang, Le Hou, Peter Grabowski, Yeqing Li, and Eugene Ie. Improving multi-agent debate with sparse communication topology. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 7281–7294, 2024b.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 17889–17904, 2024.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. Dynamic llm-agent network: An llm-agent collaboration framework with agent team optimization. *arXiv preprint arXiv:2310.02170*, 2023.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. Agentboard: An analytical evaluation board of multi-turn llm agents. *Advances in neural information processing systems*, 37:74325–74362, 2024.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of llm agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870, 2024.

Potsawee Manakul, Adian Liusie, and Mark Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 9004–9017, 2023.

John F Nash Jr. Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49, 1950.

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 O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

David Patterson, Joseph Gonzalez, Urs Hölzle, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R So, Maud Texier, and Jeff Dean. The carbon footprint of machine learning training will plateau, then shrink. *Computer*, 55(7):18–28, 2022.

Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Kunlun Zhu, Hanchen Xia, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, et al. Scaling large language model-based multi-agent collaboration. *arXiv preprint arXiv:2406.07155*, 2024.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025.

Alireza Rezazadeh, Zichao Li, Ange Lou, Yuying Zhao, Wei Wei, and Yujia Bao. Collaborative memory: Multi-user memory sharing in llm agents with dynamic access control. *arXiv preprint arXiv:2505.18279*, 2025.

Anirban Saha, Binay Gupta, Anirban Chatterjee, and Kunal Banerjee. You believe your llm is not delusional? think again! a study of llm hallucination on foundation models under perturbation. *Discover Data*, 3(1):20, 2025.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R Johnston, et al. Towards understanding sycophancy in language models. *arXiv preprint arXiv:2310.13548*, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

Andries Smit, Paul Duckworth, Nathan Grinsztajn, Thomas D Barrett, and Arnu Pretorius. Should we be going mad? a look at multi-agent debate strategies for llms. *arXiv preprint arXiv:2311.17371*, 2023.

Maojia Song, Tej Deep Pala, Ruiwen Zhou, Weisheng Jin, Amir Zadeh, Chuan Li, Dorien Herremans, and Soujanya Poria. Llms can't handle peer pressure: Crumbling under multi-agent social interactions. *arXiv preprint arXiv:2508.18321*, 2025.

Gaurang Sriramanan, Siddhant Bharti, Vinu Sankar Sadasivan, Shoumik Saha, Priyatham Kattakinda, and Soheil Feizi. Llm-check: Investigating detection of hallucinations in large language models. *Advances in Neural Information Processing Systems*, 37:34188–34216, 2024.

Theodore Sumers, Shunyu Yao, Karthik R Narasimhan, and Thomas L Griffiths. Cognitive architectures for language agents. *Transactions on Machine Learning Research*, 2023.

Luoxi Tang, Yuqiao Meng, Joseph Costa, Yingxue Zhang, Muchao Ye, and Zhaohan Xi. The value of variance: Mitigating debate collapse in multi-agent systems via uncertainty-driven policy optimization. *arXiv preprint arXiv:2602.07186*, 2026.

Qwen Team. Qwen3 technical report, 2025. URL `https://arxiv.org/abs/2505.09388`.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5433–5442, 2023.

Christian Tomani, Kamalika Chaudhuri, Ivan Evtimov, Daniel Cremers, and Mark Ibrahim. Uncertainty-based abstention in llms improves safety and reduces hallucinations. *arXiv preprint arXiv:2404.10960*, 2024.

Fei Wang, Xingchen Wan, Ruoxi Sun, Jiefeng Chen, and Sercan O Arik. Astute rag: Overcoming imperfect retrieval augmentation and knowledge conflicts for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 30553–30571, 2025a.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023a.

Haotian Wang, Xiyuan Du, Weijiang Yu, Qianglong Chen, Kun Zhu, Zheng Chu, Lian Yan, and Yi Guan. Learning to break: Knowledge-enhanced reasoning in multi-agent debate system. *Neurocomputing*, 618:129063, 2025b.

Junjian Wang, Lidan Zhao, and Xi Sheryl Zhang. Madra: Multi-agent debate for risk-aware embodied planning. *arXiv preprint arXiv:2511.21460*, 2025c.

Qineng Wang, Zihao Wang, Ying Su, Hanghang Tong, and Yangqiu Song. Rethinking the bounds of llm reasoning: Are multi-agent discussions the key? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6106–6131, 2024a.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788, 2020.

Yike Wang, Shangbin Feng, Heng Wang, Weijia Shi, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov. Resolving knowledge conflicts in large language models. *arXiv preprint arXiv:2310.00935*, 2023b.

Yu Wang and Xi Chen. Mirix: Multi-agent memory system for llm-based agents. *arXiv preprint arXiv:2507.07957*, 2025.

Zezhong Wang, Fangkai Yang, Lu Wang, Pu Zhao, Hongru Wang, Liang Chen, Qingwei Lin, and Kam-Fai Wong. Self-guard: Empower the llm to safeguard itself. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1648–1668, 2024b.

Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 257–279, 2024c.

Qianshan Wei, Tengchao Yang, Yaochen Wang, Xinfeng Li, Lijun Li, Zhenfei Yin, Yi Zhan, Thorsten Holz, Zhiqiang Lin, and XiaoFeng Wang. A-memguard: A proactive defense framework for llm-based agent memory. *arXiv preprint arXiv:2510.02373*, 2025.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First conference on language modeling*, 2024.

Yaxiong Wu, Sheng Liang, Chen Zhang, Yichao Wang, Yongyue Zhang, Huifeng Guo, Ruiming Tang, and Yong Liu. From human memory to ai memory: A survey on memory mechanisms in the era of llms. *arXiv preprint arXiv:2504.15965*, 2025.

Zhen Xiang, Linzhi Zheng, Yanjie Li, Junyuan Hong, Qinbin Li, Han Xie, Jiawei Zhang, Zidi Xiong, Chulin Xie, Carl Yang, et al. Guardagent: Safeguard llm agents by a guard agent via knowledge-enabled reasoning. *arXiv preprint arXiv:2406.09187*, 2024.

Kai Xiong, Xiao Ding, Yixin Cao, Ting Liu, and Bing Qin. Examining inter-consistency of large language models collaboration: An in-depth analysis via debate. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 7572–7590, 2023a.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. *arXiv preprint arXiv:2306.13063*, 2023b.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*, 2024.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022.

Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. G-memory: Tracing hierarchical memory for multi-agent systems. *arXiv preprint arXiv:2506.07398*, 2025a.

Hanrong Zhang, Jingyuan Huang, Kai Mei, Yifei Yao, Zhenting Wang, Chenlu Zhan, Hongwei Wang, and Yongfeng Zhang. Agent security bench (asb): Formalizing and benchmarking attacks and defenses in llm-based agents. *arXiv preprint arXiv:2410.02644*, 2024.

Jintian Zhang, Xin Xu, Ningyu Zhang, Ruibo Liu, Bryan Hooi, and Shumin Deng. Exploring collaboration mechanisms for llm agents: A social psychology view. *arXiv preprint arXiv:2310.02124*, 2023.

Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems*, 43(6):1–47, 2025b.

Zhehao Zhang, Weijie Xu, Fanyou Wu, and Chandan K Reddy. Falsereject: A resource for improving contextual safety and mitigating over-refusals in llms via structured reasoning. *arXiv preprint arXiv:2505.08054*, 2025c.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 19724–19731, 2024.

A Limitations

Our study focuses on inference-time calibration of shared memory in multi-agent debate and instantiates EquiMem on two representative memory families: embedding-based and graph-based memory. While these settings span diverse MAD frameworks, benchmarks, and memory architectures, other memory representations and deployment settings may exhibit additional interaction patterns. In addition, the effectiveness of structural calibration naturally depends on the information available in the accumulated memory and agent interactions. We provide a more detailed analysis of such edge cases and their empirical behavior in Appendix F.

B Additional Analysis of the Zero-Trust Memory Game

B.1 Existence of Memory Equilibrium

We prove Proposition 3.2. The purpose of this result is modest: it establishes that the zero-trust game has a stable solution under the finite inference budget used by a practical multi-agent system.

Proof. Fix a debate round t. The system contains a finite number of agents, so the player set $\{a_c\} \cup \mathcal{A}_t$ is finite.

The contributor operates under a bounded generation budget. Given a finite model vocabulary and a finite maximum generation length, the set of memory updates that can be produced within the round is finite. Its size can be very large, which does not affect the existence argument.

Each auditor also operates under a bounded memory-access budget. An embedding memory permits a finite number of retrieval operations within one round. A graph memory permits a finite number of traversals once the interaction budget and traversal depth are bounded. The action space of every auditor is therefore finite.

The resulting round-level interaction is a finite normal-form game. Allowing randomized actions turns each finite action set into a probability simplex. Every such simplex is non-empty, compact, and convex. Expected utilities are finite weighted sums of the utilities in Eq. 1 and are consequently continuous in the mixed strategies.

Nash's existence theorem therefore guarantees at least one mixed-strategy equilibrium. □

Proposition 3.2 does not state that every equilibrium produces reliable memory. It establishes that a stable state exists. Our method uses the structural properties of a useful equilibrium as a design objective: influence should remain coupled to consistency with the current memory, and evidence from other agents should be available at low marginal inference cost.

B.2 How Calibration Approaches the Memory Equilibrium

Equation 3 converts the qualitative equilibrium condition into an inference-time constraint. This subsection explains the connection for the contributor and auditors separately, and then describes the joint stable state produced by repeated calibration.

Contributor-side stability. Let τ denote the effective influence assigned to a proposed update. For both supported memory architectures, increasing τ increases the ability of the update to affect later retrieval. We therefore write the effective influence as $\mathbb{I}_\tau(\Delta_t, \mathcal{M}_t)$ and assume it is monotone in τ.

At the same time, EquiMem measures how much the update departs from the current memory through d_{mem} and how much it changes memory already being used through d_{use}. The feasible influence set is

$$\mathcal{T}_t = \{\tau \in [0,1] : \tau d_{\text{mem}} \le \epsilon_t,\ \tau d_{\text{use}} \le \eta_t\} \tag{6}$$

Since both deviations are non-negative, $\mathcal{T}_t$ is an interval containing zero. Equation 3 selects its largest element. Consequently, any unilateral attempt to increase the effective influence beyond τ_t violates at least one memory constraint whenever $\tau_t < 1$.

This yields the main contributor-side property.

Proposition B.1 (Maximal Feasible Influence). *Let $\mathbb{I}_\tau$ be non-decreasing in τ. The influence τ_t returned by Eq. 3 maximizes the contributor's effective influence over all updates that satisfy the current memory and memory-use constraints.*

Proof. Equation 3 directly chooses the largest element of $\mathcal{T}_t$. Monotonicity of $\mathbb{I}_\tau$ then implies that no smaller feasible value can yield greater effective influence. Any larger value lies outside $\mathcal{T}_t$ and violates at least one constraint. □

This is the operational form of the contributor equilibrium condition in Eq. 2. Once the maximum feasible influence has been assigned, the contributor cannot obtain greater memory influence while keeping the same structural consistency conditions satisfied.

Auditor-side stability. The auditor utility in Eq. 1 contains an evidence term and an effort term. EquiMem obtains the evidence term from memory actions that already took place during ordinary reasoning. For embedding memory, an auditor has already issued the retrieval query used for calibration. For graph memory, the relevant traversal path has already been visited. The additional action consists of logging this interaction and evaluating the candidate update against it.

Let c_i^{base} be the inference cost of the original reasoning step and c_i^{EquiMem} the added cost of exposing the same interaction to the calibration routine. The relevant marginal effort is

$$\Delta c_i = c_i^{\text{EquiMem}} - c_i^{\text{base}} \tag{7}$$

No new LLM generation appears in Δc_i. It contains only memory-side operations such as comparing retrieval lists or checking graph connectivity. This keeps the additional effort term small and removes the need for a separate language-model auditing step.

The memory interaction used for calibration must also occur before Δ_t becomes visible. This ordering matters because it prevents an auditor from first observing the candidate and then constructing a memory access specifically to support it. We record the latest eligible retrieval or traversal from each auditor and discard interactions that occur after the candidate enters the staging area.

Why one auditor cannot determine the calibration. The use constraint aggregates auditor-level changes with a median. Let d_i denote the change measured from auditor a_i. Then

$$d_{\text{use}} = \text{median}_{a_i \in \mathcal{A}_t}\, d_i \tag{8}$$

Changing one d_i usually leaves the median unchanged, and in all cases its effect is bounded by the neighboring order statistics. Thus a single auditor has no direct control over the memory-use constraint. More generally, when fewer than half of the observed interactions are arbitrarily altered, the median remains within the range determined by the remaining interactions. The memory-state constraint is independent of auditor behavior and remains active even when the auditor evidence is poor.

We use the median for this reason and do not maintain a separate credibility model for each agent. This keeps the mechanism independent of behavioral history and avoids introducing another state variable into the equilibrium argument.

Approximation to the ideal game. The quantities available to EQUIMEM are structural approximations to the ideal terms D and A_i in Eq. 1. Let $\widehat{D}$ and $\widehat{A}_i$ denote the quantities induced by d_{mem} and d_{use}. Suppose their approximation errors over the current feasible region satisfy

$$|D - \widehat{D}| \le \varepsilon_D \qquad |A_i - \widehat{A}_i| \le \varepsilon_A \tag{9}$$

The calibrated game therefore differs from the ideal zero-trust game only through these measurement errors and the marginal memory-side cost Δc_i. If the calibrated interaction is stable up to ξ under the measurable quantities, a unilateral deviation can improve the ideal utility by at most the calibration error added on the two sides of the comparison. In the standard approximate-equilibrium form,

$$u_j(s_j^*, s_{-j}^*) \ge u_j(s_j, s_{-j}^*) - (\xi + 2\varepsilon) \qquad \varepsilon = \max\{\varepsilon_D, \varepsilon_A, \max_i \Delta c_i\} \tag{10}$$

The argument follows directly from applying the uniform error bound once to the current action and once to the deviating action. Hence the calibrated state approaches the equilibrium of the ideal game as the structural measurements become more accurate and the marginal auditing cost decreases.

This result also clarifies the role of the two constraints. d_{mem} couples influence to the structure of the complete memory state. d_{use} couples influence to the portions of memory that other agents are currently relying on. Either source can restrict the update, so a proposal receives full influence only when both remain within their normal operating ranges.

C CALIBRATION DETAILS

C.1 EMBEDDING-BASED MEMORY

Let $\mathcal{M}_t = \{(\mathbf{e}_j, m_j)\}_{j=1}^N$ and $\Delta_t = \{(\mathbf{e}_k^\Delta, m_k^\Delta)\}_{k=1}^K$. We compute the two deviations in Eq. 3 before the candidate becomes active.

Memory deviation. For each candidate embedding $\mathbf{e}_k^\Delta$, let $\hat{r}_k$ be the number of existing memory vectors lying within radius γ_t. The radius is the running median similarity radius observed by the base retrieval index. Let $\bar{r}_t$ denote the running mean neighborhood size of existing entries. We measure only excess concentration:

$$d_{\mathrm{mem}}^{\mathrm{emb}} = \frac{1}{K} \sum_{k=1}^{K} \frac{[\hat{r}_k - \bar{r}_t]_+}{\bar{r}_t + \varepsilon} \tag{11}$$

A candidate whose neighborhood is comparable to ordinary memory entries has little deviation. A candidate located in an unusually dense region has larger deviation because it can participate in many nearby retrievals.

The positive-part operator avoids penalizing sparse additions. Such an addition may contain genuinely new information, while unusual concentration is more directly related to excessive retrieval influence.

Memory-use deviation. For every auditor with an eligible retrieval query q_i, let

$$L_i = \text{top-}k(\mathcal{M}_t, q_i) \qquad L_i^\Delta = \text{top-}k(\mathcal{M}_t \cup \Delta_t, q_i) \tag{12}$$

The first list is already cached from normal retrieval. The second is evaluated while the candidate remains staged. We compute

$$d_{\mathrm{use}}^{\mathrm{emb}} = \mathrm{median}_{i \in \mathcal{O}_t}\, d_{\mathrm{RBO}}(L_i, L_i^\Delta) \tag{13}$$

where $\mathcal{O}_t$ contains auditors with an eligible pre-update query. Rank-biased overlap gives greater weight to changes near the top of the retrieval list, which is appropriate because those entries exert the largest downstream effect.

When no eligible auditor query is available, the use constraint is omitted for that round and calibration is determined by the memory constraint. The update can be recalibrated after additional interactions become available.

Applying calibrated influence. Let $s_0(q, m_j)$ denote the native retrieval value used by the underlying memory architecture. For an entry carrying influence τ_j, EQUIMEM uses

$$s(q, m_j) = \tau_j \, s_0(q, m_j) \tag{14}$$

This form is independent of the particular similarity function used by the base memory. In particular, it does not rely on rescaling an embedding vector, which would leave cosine similarity unchanged after normalization.

C.2 GRAPH-BASED MEMORY

Let $\mathcal{M}_t = (\mathcal{V}_t, \mathcal{E}_t, \ell)$ and $\Delta_t = (\mathcal{V}_t^{\Delta}, \mathcal{E}_t^{\Delta})$. Candidate nodes and edges are first placed in a staging area and remain invisible to agents during calibration.

Relation compatibility. For two relation types r and r', let $c_t(r, r')$ be the number of nodes whose local neighborhoods contain both relations. We normalize this count by their marginal frequencies

$$\mathrm{cooc}_t(r, r') = \frac{c_t(r, r')|\mathcal{V}_t|}{(c_t(r) + \varepsilon)(c_t(r') + \varepsilon)} \tag{15}$$

Relations whose co-occurrence exceeds the running median are included in $\mathcal{R}_{\mathrm{compat}}(r)$. This set is learned entirely from the current graph and is updated as new edges become active.

Memory deviation. For a candidate edge $e = (u, r, v)$, we first measure local support

$$s_{\mathrm{local}}(e) = \frac{|\{(u, r', v') \in \mathcal{E}_t : r' \in \mathcal{R}_{\mathrm{compat}}(r)\}|}{|\mathcal{N}(u)| + \varepsilon} \tag{16}$$

We also inspect existing paths between u and v. Let $\mathcal{P}_{u,v}$ contain the paths found within the normal traversal depth of the graph memory. Their relation support is

$$s_{\mathrm{path}}(e) = \frac{1}{|\mathcal{P}_{u,v}|} \sum_{\pi \in \mathcal{P}_{u,v}} \mathbf{1}\left[\mathrm{type}(\pi) \cap \mathcal{R}_{\mathrm{compat}}(r) \neq \emptyset\right] \tag{17}$$

When no path exists, we use $s_{\mathrm{local}}(e)$ alone. Otherwise we set $\kappa_t(e) = \frac{1}{2}\left(s_{\mathrm{local}}(e) + s_{\mathrm{path}}(e)\right)$. The update-level memory deviation is

$$d_{\mathrm{mem}}^{\mathrm{graph}} = \frac{1}{|\mathcal{E}_t^{\Delta}|} \sum_{e \in \mathcal{E}_t^{\Delta}} \left(1 - \kappa_t(e)\right) \tag{18}$$

Thus edges supported by existing relation patterns produce little change, while structurally isolated edges consume more of the permitted influence budget.

Memory-use deviation. Let π_i be the most recent eligible path traversed by auditor a_i before the candidate entered staging. Define $\mathrm{connect}(e, \pi_i) = 1$ when e touches a node on π_i or connects to that path through a compatible relation. We compute

$$d_{\mathrm{use}}^{\mathrm{graph}} = \mathrm{median}_{i \in \mathcal{O}_t}\left[1 - \frac{1}{|\mathcal{E}_t^{\Delta}|} \sum_{e \in \mathcal{E}_t^{\Delta}} \mathrm{connect}(e, \pi_i)\right] \tag{19}$$

This quantity is small when the proposed update extends regions of the graph that agents are currently using. It becomes larger when the update forms a largely disconnected structure.

Applying calibrated influence. Every active edge inherits the influence τ_t of the update that created it. If the underlying graph memory assigns native strength $s_0(\pi)$ to a retrieved path π, EQUIMEM uses

$$s(\pi) = s_0(\pi) \prod_{e \in \pi} \tau_e \tag{20}$$

A path containing a weakly supported edge therefore receives proportionally less influence during later reasoning.

C.3 Adaptive Calibration Ranges

The tolerances ϵ_t and η_t in Eq. 3 describe the amount of memory change that is ordinary for the current system. We estimate them from recent proposals, which avoids task-specific threshold tuning.

Let $\mathcal{H}_t$ be a bounded history window containing the raw d_{mem} and d_{use} values observed before influence calibration. We use

$$\epsilon_t = \text{median}\left\{d_{\text{mem}}^{(\tau)} : \tau \in \mathcal{H}_t\right\} \qquad \eta_t = \text{median}\left\{d_{\text{use}}^{(\tau)} : \tau \in \mathcal{H}_t\right\} \tag{21}$$

Using raw pre-calibration deviations avoids a feedback loop in which heavily downweighted updates artificially reduce future tolerances.

During cold start, each statistic is the median of the observations available so far. The first observation therefore defines the initial operating range. As more rounds accumulate, individual early observations quickly lose influence on the median.

Because both constraints in Eq. 3 are linear in τ, the optimization has a direct solution whenever the corresponding deviations are positive

$$\tau_t = \min\left\{1,\ \frac{\epsilon_t}{d_{\text{mem}} + \varepsilon},\ \frac{\eta_t}{d_{\text{use}} + \varepsilon}\right\} \tag{22}$$

When no auditor interaction is available, the third term is omitted. This closed form means that calibration requires no iterative optimization at inference time.

C.4 Discussion: How EquiMem Handles the Motivating Cases

The three cases in Figure 1 differ in which agents appear confident, but they share the same underlying problem: the decision about shared memory is too easily determined by the agents' expressed judgments. EquiMem does not treat these cases separately. Instead, it applies the same procedure to every proposed update: the update is first evaluated against the existing memory and against the memory already being used by other agents, and its subsequent influence is calibrated from these two checks. Agent confidence, agreement, or disagreement does not directly determine this calibration.

Confident contributor, uncertain auditors. When an incorrect contributor is confident and the other agents fail to challenge it, EquiMem does not interpret their lack of objection as evidence that the update is reliable. If the proposed update conflicts with the accumulated memory or substantially changes information that auditors have already retrieved or traversed, its influence is reduced even when no auditor explicitly raises the conflict. Thus, a contributor cannot gain greater memory influence simply by sounding more certain than the other agents.

Confident auditors, tentative contributor. The converse case is handled in the same way. Auditors do not have a direct veto over a proposed update, and their stated confidence is not used as a calibration signal. A tentatively expressed update can therefore retain substantial influence when it agrees with the existing memory and does not disrupt the evidence already being used by other agents. The mechanism depends on the update's relation to memory, rather than on which side of the debate sounds more confident.

All agents are uncertain. Likewise, uncertainty among all participants need not force a proposed update to be discarded. Once an update is proposed, EquiMem can keep it in memory with an influence determined by the same two checks. An update that is well supported can remain influential despite uncertain language, while a poorly supported one is retained with reduced influence. This separates whether information is stored from how strongly it affects later retrieval. Of course, EquiMem cannot recover useful information that no agent proposes in the first place; its role is to calibrate proposed memory updates rather than generate missing ones.

Taken together, these cases illustrate why EquiMem operates below the level of debate confidence or voting. The same calibration rule applies regardless of whether confidence lies with the contributor, with the auditors, or with neither side. What changes across cases is the behavior of the agents; what remains fixed is how EquiMem determines the influence of the resulting memory update.

D INFERENCE-TIME EFFICIENCY

EQUIMEM adds memory-side computation while leaving the agent reasoning process unchanged. This distinction is important because language-model generation dominates the token and latency cost of the multi-agent system.

Embedding memory. The neighborhood statistic in Eq. 11 is obtained from the nearest-neighbor index already maintained by the memory architecture. No second index is required.

For the memory-use constraint, the pre-update list L_i has already been returned to auditor a_i during normal reasoning. We cache this result. To construct L_i^{Δ}, the implementation evaluates the small set of candidate embeddings against q_i and merges them with the cached top-k entries. A complete second search over $\mathcal{M}_t$ is therefore unnecessary.

The influence value τ_j is stored as one scalar with each entry. During retrieval, it is applied to the native retrieval value after the candidate set has been obtained. The underlying embedding representation and nearest-neighbor data structure remain unchanged.

Graph memory. The local relation check reads adjacency information already stored by the graph memory. Relation co-occurrence counts are updated incrementally after an edge becomes active. Existing paths used for Eq. 17 can be taken from the graph retrieval process when available; a bounded search is performed only when the relevant endpoint pair has no cached path information.

Auditor paths used in Eq. 19 are recorded from normal traversals before the update is staged. Candidate edges remain in a temporary buffer during calibration, so this procedure does not require a second visible graph.

The influence value of each committed edge is stored as an edge attribute. Path retrieval accumulates these values together with the native path strength. No auxiliary graph is maintained.

No additional language-model generation. The calibration procedure never asks an agent to judge an update, explain a decision, report confidence, or perform a separate verification turn. Consequently, EQUIMEM introduces no calibration prompt and no additional generated tokens. The added work consists of memory-index lookup, cached result comparison, graph lookup, and scalar arithmetic.

Persistent state. For embedding memory, the main additional state is one τ value per stored entry and a bounded history of recent deviation measurements. For graph memory, each edge carries the same scalar value, together with the relation co-occurrence table used by $\mathcal{R}_{\mathrm{compat}}$. These quantities are updated online with the underlying memory.

E EXPERIMENTS RESULTS ANALYSIS

E.1 CONFIGURATION

LLM backbone and implementation. All experiments in Table 1 use Qwen3-VL-8B-Instruct (Team, 2025) as the agent backbone, deployed locally via Ollama with $N = 6$ agents per system. We select Qwen3-VL-8B-Instruct because it achieves SOTA on AgentBench and ToolBench among 8B models under the Apache 2.0 license, while its native function-calling interface simplifies multi-agent orchestration. Alternative backbones (DeepSeek-R1-Distill-7B, Llama-3.1-8B) are evaluated in Appendix E.5.

Each agent operates independently with no shared parameters or centralised coordination, except for interaction through the shared memory $\mathcal{M}$. EQUIMEM's calibration is implemented as a standalone Python process with no generative LLM dependency at runtime. It computes the memory deviation d_{mem}, use deviation d_{use}, and calibrated influence τ_t from observable memory state and existing agent interactions, consistent with the zero-trust setting in Definition 3.1.

Embedding-space computations for $d_{\mathrm{mem}}^{\mathrm{emb}}$ and $d_{\mathrm{use}}^{\mathrm{emb}}$ use all-MiniLM-L6-v2 (Wang et al., 2020) with approximate nearest-neighbour retrieval for efficient scaling to large memory sizes. Experiments with varying agent counts $N \in \{4, 10\}$ are reported in Appendix E.6.

Hyperparameter settings. We summarise the key hyperparameters in Table 2. Unlike prior multi-agent or memory-based systems that rely on manually tuned thresholds, EQUIMEM minimises the number of fixed hyperparameters by estimating most quantities directly from data.

Table 2: Key hyperparameters and configurations in EQUIMEM.

Component	Parameter	Value / Strategy
Agents	Number of agents N	6 (default), varied in Appendix
LLM	Backbone	Qwen3-VL-8B-Instruct
Memory	Top-k retrieval	$k = 5$
Embedding	Dimension d	384 (MiniLM default)
Calibration	Operating ranges ϵ_t, η_t	Running medians of raw deviations
Embedding	Similarity radius	Median similarity (adaptive)
Graph	Path depth $L_{\max}$	$\lceil \log \lvert\mathcal{V}\rvert \rceil$
Graph	Relation compatibility	Online co-occurrence statistics
Use deviation	Auditor aggregation	Median over eligible interactions

Adaptive parameter estimation. A key design principle of EQUIMEM is that most hyperparameters are *data-adaptive* rather than manually tuned:

- The operating ranges ϵ_t and η_t are running medians of recent raw d_{mem} and d_{use} values, eliminating task-specific tuning of the influence calibration.
- The embedding similarity radius is derived from the running median of observed similarities, while graph relation compatibility is estimated from historical relation co-occurrence patterns.
- The use deviation aggregates eligible auditor retrievals or traversals with a median, requiring no auditor-specific reliability parameter.
- Graph traversal depth $L_{\max}$ scales with the logarithm of graph size, reflecting typical structural properties of real-world knowledge graphs while maintaining computational efficiency.

This adaptive design serves two purposes. First, it reduces the risk of overfitting hyperparameters to specific benchmarks, improving generalisation across tasks and memory architectures. Second, it aligns with the zero-trust assumption by avoiding reliance on any externally tuned or agent-provided signals. All thresholds are grounded in observable system statistics, ensuring that calibration remains consistent and robust under adversarial or non-cooperative agent behaviours.

E.2 DATASET STATISTICS

Why these benchmarks. The four benchmarks are selected to stress-test the primary failure modes identified by our zero-trust game analysis: (i) a single dominant contributor injecting a locally plausible but globally inconsistent entry, and (ii) auditors rubber-stamping an incorrect proposal when uncertain or sycophantic. These failures cause particular harm in long-horizon tasks-premature goal states in embodied action, fabricated intermediate facts in multi-hop reasoning, and temporal contradictions in lifelong conversation. The selected datasets span knowledge-intensive and action-intensive domains, providing the diverse challenge needed to evaluate EQUIMEM's calibration.

Table 3 summarises the four benchmarks used in our experiments. We report the number of task instances, the average interaction length, the evaluation metric, and the domain of each benchmark.

ALFWorld is a text-based embodied environment derived from ALFRED, covering six task categories: `pick_and_place`, `look_at_obj_in_light`, `pick_clean_then_place`, `pick_heat_then_place`, `pick_cool_then_place`, and `pick_two_obj_and_place`. We use the `valid_unseen` split (134 tasks), where each task requires a sequence of physical actions with an average length of approximately 25 steps. **Why it matters:** The long action chain and strict precondition dependencies make this benchmark particularly susceptible to premature goal-state injection-a corrupted memory entry that

Table 3: Benchmark statistics for the four evaluation benchmarks.

Benchmark	#Tasks	Avg. Len.	Metric	Domain	Source
ALFWorld	134	$\sim$25 steps	Success rate	Embodied	Shridhar et al. (2020)
Webshop	1000	$\sim$30 steps	Reward	Embodied	Yao et al. (2022)
HotpotQA	500	1 query	Accuracy	Knowledge	Yang et al. (2018)
LoCoMo	500	$<$300 turns	F1	Knowledge	Maharana et al. (2024)

incorrectly asserts a task is complete can terminate the entire episode with zero reward (see Case 2 in Appendix F).

WebShop is an embodied decision-making benchmark that simulates online shopping through natural language interactions. Each task requires an agent to navigate search results, apply filters, and inspect product pages to locate a specified item. We evaluate on 1,000 randomly sampled tasks, with an average trajectory length of approximately 30 steps. **Why it matters:** The large, dynamic search space forces frequent memory writes under uncertainty; a dominant agent that confidently commits an incorrect product attribute (e.g., a wrong color or price) can mislead the entire downstream search process.

HotpotQA is a multi-hop question answering benchmark constructed from Wikipedia. Each instance requires reasoning across multiple documents to derive the final answer. We use 500 questions drawn from the full HotpotQA dev set, each consisting of a single query, and report accuracy. **Why it matters:** A hallucinated intermediate fact committed to shared memory can override correct evidence later retrieved by other agents, directly testing whether d_{use} captures inconsistent overwrites in the retrieval pool (see Case 1).

LoCoMo is a long-context conversational memory benchmark. Each original dialogue spans up to 35 sessions and averages 300 turns, simulating a lifelong, multi-session human conversation. From these long-form dialogues, we sample 500 question-answering tasks, each derived from a *separate* dialogue to guarantee statistical independence. Every task presents the full dialogue history in memory and asks one evaluative question requiring cross-session temporal reasoning. We report F1 as the primary evaluation metric. **Why it matters:** The extreme temporal span (300 turns) naturally creates memory decay, contradiction, and topic drift across sessions. This tests the most dangerous failure mode for our theory: a contributor who proposes a confident but incorrect summary, and a room of auditors who, due to the cost of searching 300-turn memory, prefer to agree rather than verify (see Case 3).

E.3 Extended Analysis of Overall Effectiveness

This section expands on our insights of §5.2 with deeper mechanistic explanation.

Why isolated checks saturate? On embedding memory, the gain from Vanilla to the strongest per-entry baseline (A-MemGuard) is modest, and EquiMem then adds a further substantial improvement on top, exceeding the combined gain of all three intermediate baselines (LLM Audit, PPL Filter, A-MemGuard). The three baselines also cluster tightly with frequent rank inversions, suggesting they capture overlapping signals rather than complementary ones.

Both observations are consistent with the saturation hypothesis: isolated entry level checks evaluate only the *intrinsic quality* of Δ_t (grammaticality, fluency, perplexity), while the dominant errors in a growing memory $\mathcal{M}_t$ arise from *relational inconsistency*, where an entry reads perfectly in isolation yet contradicts existing memory. EquiMem breaks this ceiling by constraining influence through both d_{mem}, which measures deviation from the current memory structure, and d_{use}, which measures disruption to memory already used by other agents.

Qualitative pattern of ceiling aware gains. An inverse scaling effect is visible across all configurations: EquiMem's gain over the next best baseline is largest where that baseline is weakest, and shrinks where the baseline already performs well. This pattern holds consistently across reasoning intensive and action intensive benchmarks alike, with the largest improvements appearing in settings where existing safeguards struggle the most.

This shrinkage follows from continuous influence calibration: when the incoming updates already remain close to the normal memory and use ranges, τ_t stays near one and EquiMem makes only

Table 4: Ablation study of EQUIMEM's calibration components (%). We ablate one component at a time, following the notation in Section 4. For embedding memory, we remove the memory deviation d_{mem} or memory-use deviation d_{use} directly. For graph memory, we additionally ablate the local and path consistency terms c_{local} and c_{path} that constitute d_{mem}. We also remove calibrated influence to test the effect of applying τ_t during subsequent retrieval. All configurations use AutoGen + `Qwen3-VL-8B-Instruct`, $N = 6$ agents, in the benign setting (mean±std over 5 runs).

		AutoGen				MacNet				DyLAN			
Memory	**Ablation**	**HQA**	**LCM**	**ALF**	**WS**	**HQA**	**LCM**	**ALF**	**WS**	**HQA**	**LCM**	**ALF**	**WS**
MemBank (emb.)	w/o d_{mem}	46.8±6.9	33.8±6.1	74.0±3.6	33.9±7.9	44.6±9.2	30.4±10.0	56.2±1.0	45.0±6.6	42.0±0.9	32.9±5.8	58.3±2.8	36.7±3.5
	w/o d_{use}	38.4±5.3	23.7±6.9	67.9±4.3	35.1±7.5	48.8±7.5	21.6±10.2	59.9±1.1	46.7±7.7	39.1±0.8	26.1±7.1	65.5±2.6	43.4±3.8
	w/o calibrated influence	49.7±6.3	34.9±5.5	71.6±3.7	40.1±7.7	51.1±8.0	32.3±11.4	65.2±1.3	44.8±5.8	53.6±0.8	36.7±7.0	65.6±2.6	47.5±3.5
G-Memory (graph)	w/o c_{local} in d_{mem}	42.9±10.5	57.9±3.7	77.0±3.7	53.3±6.2	42.6±8.0	52.9±1.6	66.8±9.3	55.8±4.0	45.9±8.2	45.8±2.8	59.9±11.4	46.1±6.4
	w/o c_{path} in d_{mem}	36.5±10.1	46.8±3.9	74.6±3.7	47.6±6.3	33.2±6.2	46.3±1.8	75.8±8.7	58.1±3.3	35.0±8.3	39.0±2.8	71.4±11.5	46.9±6.7
	w/o d_{use}	39.2±10.4	51.2±3.2	77.7±3.6	54.3±6.8	39.3±6.5	54.8±1.5	62.7±9.8	50.4±3.4	36.5±8.3	41.2±2.4	61.4±11.9	45.8±5.6
	w/o calibrated influence	50.9±10.0	56.3±3.2	80.5±4.1	50.2±6.2	42.3±6.8	52.2±1.7	67.2±10.4	50.6±3.1	49.6±9.3	51.0±2.7	68.6±10.7	52.3±6.6

a small change. Importantly, this inverse scaling pattern is *inconsistent* with the hypothesis that performance gains stem from external knowledge leakage, which would produce a constant additive boost regardless of baseline strength. Instead, it is *consistent* with EQUIMEM limiting the influence of updates that conflict with the accumulated memory state.

Memory benchmark interaction. The strongest results for reasoning intensive benchmarks come from AriGraph paired with EQUIMEM, while the strongest action task results come from G-Memory paired with EQUIMEM. This complementarity is already visible in the Vanilla setting: AriGraph leads on multi hop QA, while G-Memory leads on embodied action and long dialogue tasks. These pre existing biases reflect each architecture's native design. AriGraph's semantic knowledge graph with episodic memory is well suited to multi hop QA, while G-Memory's three tier hierarchical structure captures the long action chains and multi session dialogue history of action and conversational tasks.

EQUIMEM's calibration *inherits and amplifies* these biases rather than imposing a generic operation that flattens architectural differences. The deviations d_{mem} and d_{use} are instantiated from each architecture's geometry (embedding neighbourhoods and retrieval changes) or topology (graph neighbourhoods, paths, and traversals), so noisy updates receive less downstream influence while the architecture's native signal is preserved. The gap between AriGraph and G-Memory therefore widens under EQUIMEM in the directions each architecture is naturally suited to, rather than narrowing toward a common baseline.

Cross MAS consistency. Absolute performance differs substantially across MAD frameworks, with AutoGen generally leading, followed by DyLAN and MacNet. Yet EQUIMEM's gain over the next best baseline is comparably stable across all three frameworks. The same ranking and the same inverse scaling pattern persist regardless of which orchestration scheme is used.

The stability of the gain across orchestration regimes confirms that EQUIMEM acts at the memory layer rather than at the coordination layer. Weak orchestration and strong orchestration receive similar lifts because calibration assigns τ_t when Δ_t enters memory, upstream of any MAS-specific aggregation. This decoupling is a practical advantage: EQUIMEM can be deployed with any MAD framework without re-tuning or framework-specific adaptation.

E.4 ABLATION STUDY

This part dissects EQUIMEM's calibration into its main measurements to identify which elements drive the gains observed in RQ1. We ablate one component at a time on two representative memory architectures (MemBank and G-Memory) across all three MAS frameworks and four benchmarks. Table 4 reports the results.

Both memory-state and memory-use deviations contribute materially. Removing either deviation consistently reduces performance, showing that the two constraints capture complementary aspects of update quality. On embedding memory, removing d_{mem} eliminates the constraint that detects unusually concentrated additions to the existing representation, while removing d_{use} eliminates the signal measuring disruption to retrievals already used by other agents. The latter produces particularly large drops on several reasoning-intensive configurations, indicating that preserving the memory

Table 5: Performance with alternative LLM backbones (%) on MacNet, $N = 6$ agents, benign setting.

		Qwen3-VL-8B-Instruct (primary)				DeepSeek-R1-Distill-7B				Llama-3.1-8B			
Memory	**Safeguard**	**HQA**	**LCM**	**ALF**	**WS**	**HQA**	**LCM**	**ALF**	**WS**	**HQA**	**LCM**	**ALF**	**WS**
MemBank (emb.)	Vanilla	34.0±7.9	21.1±11.0	49.1±5.2	21.3±8.3	35.4±7.9	25.1±10.0	45.5±4.6	25.1±5.1	31.0±8.9	17.6±7.7	46.6±2.1	22.0±7.2
	+ LLM Audit	38.5±9.2	22.2±12.1	62.3±5.6	31.2±5.7	46.4±8.1	26.5±12.0	60.2±3.4	30.9±8.5	38.5±9.4	19.4±9.5	55.7±6.0	28.9±6.3
	+ PPL Filter	39.8±10.2	27.9±10.5	64.0±1.7	25.8±9.3	35.5±7.8	34.0±12.9	55.4±7.9	23.2±7.1	34.8±10.9	21.3±12.7	56.3±3.1	24.7±7.9
	+ A-MemGuard	40.7±7.1	33.9±7.6	56.5±3.7	43.4±6.2	45.7±5.9	34.0±7.4	48.8±4.8	37.3±6.9	46.2±7.8	27.6±9.9	51.7±3.8	32.9±7.9
	+ EQUIMEM	**62.3**±7.7	**35.4**±10.4	**71.3**±6.3	**56.9**±5.8	**46.9**±6.8	**34.0**±11.1	**60.2**±4.8	**50.0**±7.1	**46.2**±8.0	**32.1**±7.7	**56.3**±3.7	**45.2**±7.4
G-Memory (graph)	Vanilla	34.8±8.6	50.1±1.6	71.3±13.0	48.8±5.3	42.6±6.2	53.2±1.7	63.6±13.2	38.8±6.2	29.3±6.8	43.3±1.4	61.5±13.2	40.0±4.4
	+ LLM Audit	38.5±8.4	48.0±1.7	71.5±12.1	48.5±5.9	40.1±6.1	53.2±1.8	63.6±11.6	42.4±4.9	33.6±7.6	50.1±1.4	62.3±12.1	42.9±3.8
	+ PPL Filter	39.4±5.5	52.9±1.1	68.0±10.2	59.5±3.5	46.7±7.3	52.0±1.3	63.6±11.5	46.6±3.1	39.9±7.2	45.7±1.7	60.3±10.4	48.7±4.1
	+ A-MemGuard	—	—	—	—	—	—	—	—	—	—	—	—
	+ EQUIMEM	**50.7**±6.6	**58.6**±1.7	**72.3**±6.9	**67.6**±2.7	**52.9**±6.8	**53.2**±1.4	**63.6**±9.5	**60.0**±3.9	**44.7**±8.5	**58.5**±1.3	**60.3**±7.0	**57.6**±3.1

context actively supporting the debate is important beyond assessing the candidate against the memory state alone.

Graph calibration benefits from both local and path-level structural evidence. For graph memory, ablating either c_{local} or c_{path} weakens the construction of d_{mem} and causes substantial degradation across benchmarks. The two terms provide distinct structural evidence: c_{local} measures whether the proposed relation is compatible with the neighborhoods around its endpoints, whereas c_{path} measures whether existing multi-hop structure supports the proposed relation. Removing d_{use}, which measures compatibility with graph regions already traversed by auditors, produces additional losses. Together, these results support the shared formulation in Section 4: calibration benefits from considering both how an update changes the memory itself and how it interacts with memory already supporting other agents' reasoning.

Calibrated influence is necessary to convert deviation measurements into downstream protection. Removing calibrated influence while retaining the underlying deviation measurements also reduces performance across most configurations. This ablation leaves the system able to measure d_{mem} and d_{use} but prevents those signals from regulating subsequent retrieval through τ_t. The result confirms that the benefit of EQUIMEM does not come only from detecting atypical updates; it depends on carrying the resulting calibration forward so that weakly supported entries exert proportionally less influence on later reasoning.

Auditor-side aggregation has the smallest benign effect. The ablation that removes the reweighting mechanism causes the smallest drops in most benign configurations. This is expected because benign retrievals and traversals usually provide compatible evidence, so the robust aggregation differs little from a non-robust aggregate. Its value becomes more visible under Audit Collusion (Section 5.3), where a subset of adversarial interactions can be deliberately steered.

No single component is sufficient. Even the most impactful individual ablation does not collapse performance to the Vanilla level. The ablated system always retains a clear margin above the corresponding Vanilla score, showing that detection, alignment, and reweighting provide complementary protection. The full method performs best when all calibration signals jointly constrain the calibrated influence τ_t.

E.5 ALTERNATIVE LLM BACKBONES

Table 1 uses Qwen3-VL-8B-Instruct as the agent backbone throughout. To verify that EQUIMEM's advantage is not specific to one LLM, we replicate the MacNet evaluation on MemBank and G-Memory with two additional 7B/8B-class open-source models: **DeepSeek-R1-Distill-7B** (Guo et al., 2025), a reasoning-optimised distillation of DeepSeek-R1, and **Llama-3.1-8B-Instruct** (Patterson et al., 2022), a general-purpose instruction-tuned model. Table 5 reports the results.

EQUIMEM ranks first across all backbones. Across all memory, benchmark, and backbone combinations, EQUIMEM achieves the highest score. The relative ordering among safeguards is largely preserved, with EQUIMEM consistently outperforming existing safeguards, while intermediate baselines show occasional variations across settings as observed in Table 1.

Backbone strength shifts absolute levels but not relative patterns. Qwen3-VL-8B-Instruct generally achieves higher absolute performance among the three backbones, followed by DeepSeek-

R1-Distill-7B and Llama-3.1-8B. The gap is consistent across safeguards: weaker backbones produce lower scores for most methods, not just for EQUIMEM. Crucially, EQUIMEM's *relative* gain over the best baseline remains substantial across all backbones, confirming that calibration provides consistent benefit regardless of backbone.

Calibration remains effective across backbone capabilities. The interaction between backbone and calibration shows a consistent pattern: stronger backbones produce higher-quality memory entries, which in turn provide EQUIMEM richer memory structure and memory-use evidence for calibration. Across different backbones, EQUIMEM maintains strong performance improvements, confirming that it amplifies backbone capability rather than substituting for it: calibration is complementary to, not independent of, the underlying LLM's capability.

E.6 EFFECT OF AGENT COUNT

The main text reports results for $N = 6$ agents. This appendix evaluates how EQUIMEM's effectiveness varies with the number of participating agents. We test $N \in \{4, 6, 10\}$ on the AutoGen framework with Qwen3-VL-8B-Instruct and the G-Memory backbone in the benign setting, where $N = 6$ corresponds to the default configuration used in the main experiments.

Table 6: Effect of agent count N on benign-setting performance (AutoGen, Qwen3-VL-8B-Instruct, G-Memory backbone). Values are reported in percentage (mean±std over 5 runs). Δ denotes the gain of EQUIMEM over the next-best safeguard under the same agent count.

Benchmark	N	Vanilla	LLM Audit	PPL Filter	EQUIMEM	Δ
HotpotQA	4	33.9±9.4	36.5±8.9	42.1±8.4	**50.5**±8.0	+8.4
	10	36.8±8.8	39.4±8.3	45.6±7.9	**53.6**±7.5	+8.0
LoCoMo	4	41.5±4.5	47.6±4.8	51.0±4.5	**57.1**±3.4	+6.1
	10	44.1±3.9	50.7±4.3	54.0±4.0	**60.0**±2.8	+6.0
ALFWorld	4	70.4±4.1	74.9±3.5	76.8±3.3	**82.4**±3.5	+5.6
	10	73.0±3.6	77.4±3.0	79.1±2.9	**85.1**±3.0	+6.0
WebShop	4	37.8±6.6	39.5±6.4	46.1±6.3	**52.1**±6.0	+6.0
	10	40.4±6.1	42.0±5.8	49.4±5.9	**54.7**±5.5	+5.3

Small agent counts ($N = 4$). With only four agents, each update is evaluated using a limited number of eligible auditor memory interactions. Despite this reduced diversity, EQUIMEM consistently outperforms all baselines by a substantial margin across benchmarks. Compared to the next-best safeguard, the absolute gains remain stable across benchmarks, indicating that EQUIMEM does not rely on large numbers of auditors to function effectively. However, we observe slightly higher variance at $N = 4$, because calibration signals are estimated from fewer auditor interactions.

Moderate agent counts ($N = 6$). At $N = 6$, performance improves modestly across all methods, with the largest gains observed when moving from smaller agent counts to the default setting. This suggests that increasing the number of agents initially provides more diverse reasoning traces and memory interactions, improving both memory quality and calibration accuracy. Notably, EQUIMEM achieves strong overall performance at this setting, balancing interaction diversity and coordination overhead.

Larger agent counts ($N = 10$). Increasing the agent count further to $N = 10$ yields only marginal improvements over $N = 6$. While calibration benefits from more diverse auditor interactions, these gains are offset by increased redundancy among agent reasoning traces and memory accesses. As a result, performance begins to saturate, indicating diminishing returns from additional agents. Importantly, EQUIMEM maintains a consistent advantage over all baselines, suggesting that its effectiveness is largely insensitive to the exact choice of N.

Scaling implications. Overall, agent count primarily affects the *quality of collective memory interactions* rather than acting as a dominant performance driver. EQUIMEM requires no assumption about a minimum number of reliable agents (Definition 3.1) and remains effective even at small N. Increasing N improves stability and slightly enhances performance, but the benefits quickly plateau

beyond $N = 6$. For very large agent pools ($N > 50$), eligible auditor retrievals or traversals can be sub-sampled to a random subset of $\min(N, 20)$ agents with negligible impact on calibration quality, ensuring scalability without increasing latency.

E.7 Effect of Debate Rounds (RQ1 Supplement)

Figure 5 examines how performance evolves as the number of debate rounds increases from 1 to 5, using LoCoMo (F1) on MacNet + Qwen3-VL-8B-Instruct with $N = 6$ agents.

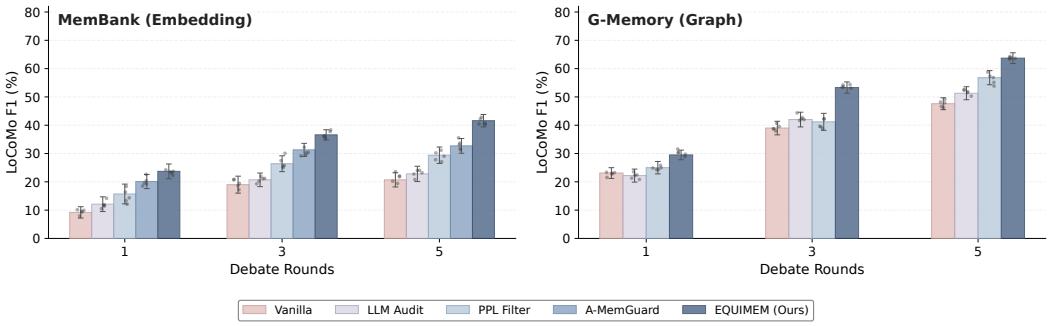


Figure 5: LoCoMo F1 vs. debate rounds on two memory architectures (MacNet, Qwen3-VL-8B-Instruct, $N = 6$).

Embedding memory saturates early. On MemBank (left panel), all methods show a sharp rise from round 1 to round 3, followed by near-zero improvement from round 3 to round 5. This saturation occurs because embedding memory accumulates dense vectors that increasingly overlap, reducing the marginal information content of later rounds. EquiMem follows the same saturation shape but remains at a higher performance level throughout all rounds, with the gap over Vanilla widening at later rounds. The baselines (LLM Audit, PPL Filter, A-MemGuard) also saturate, clustering between Vanilla and EquiMem.

Graph memory benefits from longer debate. On G-Memory (right panel), all methods improve steadily across all debate rounds. Vanilla shows consistent gains as the number of rounds increases, while EquiMem further amplifies this trend. G-Memory's hierarchical knowledge graph continues to benefit from additional debate rounds because each round's interactions extend the graph's coverage into previously unreached regions of the reasoning space. As a result, EquiMem maintains an increasing advantage over Vanilla at later rounds.

Calibration benefit grows with rounds. In both panels, the gap between EquiMem and baselines tends to widen at later rounds. This is expected: as more entries accumulate in $\mathcal{M}_t$, the fraction of locally plausible but globally inconsistent entries grows, and EquiMem's detection and alignment calibration becomes increasingly valuable. Isolated entry-level checks (LLM Audit, PPL Filter, A-MemGuard) cannot leverage the growing memory state and therefore plateau alongside Vanilla.

E.8 Adversarial Baseline Comparison

This appendix supplements Section 5.3 by reporting the adversarial performance of LLM Audit and PPL Filter under the same two attack schemes, memory architectures, benchmarks, and adversary counts used for EquiMem in the main text. All experiments use MacNet + Qwen3-VL-8B-Instruct with $N = 6$ agents. Algorithms 1 and 2 detail the two adaptive attack procedures used throughout this section and Section 5.3.

Attack settings. As specified in Algorithms 1-2, both attacks follow the *adaptive* threat model: adversarial agents have full knowledge of the safeguard mechanism they face and craft their poisoned entries or colluding memory interactions accordingly. For Memory Poisoning, adversarial contributors generate low-perplexity, fluent text that passes PPL Filter's threshold and receive high scores from LLM Audit's auxiliary judgment (since both defences evaluate entries in isolation, the attacker simply ensures each poisoned entry reads well on its own). For Audit Collusion, adversarial auditors coordinate their retrievals or traversals to make the poisoned delta appear less disruptive under

Algorithm 1 Memory Poisoning Attack

Require: Adversary set $\mathcal{K} \subset \mathcal{N}$, memory state $\mathcal{M}_t$, target safeguard type
Ensure: Poisoned delta Δ_t^{poison}

for each round t where contributor $a_c \in \mathcal{K}$ **do**
  % - Craft poison adapted to safeguard -
  if $\mathcal{M}_t$ is embedding-based **then**
    $\boldsymbol{\mu} \leftarrow \texttt{Centroid}(\{\mathbf{e}_j\}_{j=1}^{N})$ ▷ centroid of existing vectors
    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ ▷ small perturbation
    $\mathbf{e}^{\text{poison}} \leftarrow (\boldsymbol{\mu} + \boldsymbol{\epsilon}) \,/\, \|\boldsymbol{\mu} + \boldsymbol{\epsilon}\|$ ▷ retrieval magnet
    $m^{\text{poison}} \leftarrow \texttt{LLM}(\texttt{"generate fluent text near:"}, \mathcal{M}_t)$ ▷ low PPL
    $\Delta_t^{\text{poison}} \leftarrow \{(\mathbf{e}^{\text{poison}},\, m^{\text{poison}})\}$
  end if
  if $\mathcal{M}_t$ is graph-based **then**
    $u \leftarrow \arg\max_{v \in \mathcal{V}_t} \deg(v)$ ▷ high-degree node
    $v \leftarrow \texttt{DistantNode}(u,\, \mathcal{M}_t,\, \text{min_hops}{=}3)$ ▷ ≥ 3 hops away
    $r \leftarrow \texttt{MostCommonRelation}(u,\, \mathcal{M}_t)$ ▷ locally plausible
    $\Delta_t^{\text{poison}} \leftarrow \{(u,\, r,\, v)\}$ ▷ high c_{local}, low c_{path}
  end if
  % - Adaptive: ensure poison evades isolated checks -
  Assert $\texttt{PPL}(m^{\text{poison}}) < \tau_{\text{PPL}}$ ▷ defeats PPL Filter
  Assert $\texttt{LLM_Score}(\Delta_t^{\text{poison}}) > 0.5$ ▷ defeats LLM Audit
  return Δ_t^{poison}
end for

d_{use}. This has no effect on LLM Audit or PPL Filter, since neither baseline uses auditor memory interactions. We test small, medium, and large adversary counts: $k = 1, 2, 3$ out of $N = 6$ agents (17%, 33%, 50% of the group). Adversary identities are fixed per run and randomised across the 5 seeds. The same adaptive attack configurations are used for EQUIMEM in Section 5.3, ensuring a fair comparison: the attacker adapts to whichever defence is deployed.

Both baselines collapse under attack. Figure 6 reveals a consistent pattern: both LLM Audit and PPL Filter degrade roughly linearly with the number of adversaries, with no sign of recovery or stabilisation. With half the agents adversarial ($k = 3$), LLM Audit loses a substantial fraction of its benign performance on both embedding and graph memories, dropping to levels far below the no-memory baseline. PPL Filter follows a similar trajectory, suffering losses of comparable magnitude. In contrast, under the same attack conditions, EQUIMEM retains the vast majority of its benign performance, outperforming each baseline by a large margin.

Baselines fall below the Stateless level. The Stateless baseline (no memory, no safeguard) provides a simple reference: any memory system that performs worse than this has been turned into a liability. At the highest adversary count, LLM Audit and PPL Filter consistently score *below* the Stateless level on several benchmarks. This means the attacker has successfully reversed the benefit of memory. By contrast, EQUIMEM never falls below the Stateless level in any configuration, even when half the agents are adversarial.

Why baselines lack resilience. The root cause is that LLM Audit and PPL Filter evaluate each Δ_t in isolation, without measuring its consistency with the current memory or its effect on memory already used by other agents.

LLM Audit scores each proposed delta by prompting an auxiliary LLM with the delta text and generating a judgment. A poisoned entry that reads as fluent, well-structured text receives a high audit score regardless of whether it is consistent with $\mathcal{M}_t$. Under Audit Collusion, the attack is even simpler: the adversarial contributor crafts a plausible-sounding delta, and the colluding auditors are irrelevant because LLM Audit does not use auditor memory interactions at all; it relies solely on its own LLM call. The result is that Memory Poisoning and Audit Collusion produce nearly identical degradation curves for LLM Audit (top-left vs. top-right in Figure 6), confirming that LLM Audit's vulnerability is to the *content* of poisoned entries, not to the coordination of adversaries.

Algorithm 2 Audit Collusion Attack

Require: Adversary set $\mathcal{K} \subset \mathcal{N}$ ($|\mathcal{K}| = k$), poisoned delta Δ_t from colluding contributor, memory state $\mathcal{M}_t$, target safeguard type
Ensure: Modified auditor memory interactions $\{x_i^{\text{collude}}\}_{a_i \in \mathcal{K} \cap \mathcal{A}_t}$

for each round t where auditors $\mathcal{K} \cap \mathcal{A}_t \neq \emptyset$ **do**
  % - Colluding contributor injects mild poison -
  $\Delta_t \leftarrow \texttt{CraftDelta}(a_c, \mathcal{M}_t)$ ▷ plausible but incorrect
  % - Each colluding auditor steers a supportive memory interaction -
  for each $a_i \in \mathcal{K} \cap \mathcal{A}_t$ **do**
    if $\mathcal{M}_t$ is embedding-based **then**
      $\mathbf{e}^{\text{far}} \leftarrow \arg\min_{\mathbf{e}_j \in \mathcal{M}_t} \cos(\mathbf{e}_j, \mathbf{e}^{\Delta})$ ▷ farthest from poison
      $q_i^{\text{collude}} \leftarrow \texttt{TextOf}(\mathbf{e}^{\text{far}})$
      $x_i^{\text{collude}} \leftarrow q_i^{\text{collude}}$ ▷ top-k won't include Δ_t
      % Result: d_{RBO}(top-$k(\mathcal{M}_t, q_i)$, top-$k(\mathcal{M}_t \cup \Delta_t, q_i)) \approx 0$
    end if
    if $\mathcal{M}_t$ is graph-based **then**
      $(u, r, v) \leftarrow$ sampled edge from Δ_t
      $\pi_i^{\text{collude}} \leftarrow \texttt{BuildPathTo}(u, \mathcal{M}_t, \mathcal{R}_{\text{compat}})$
      % Fabricate a path that reaches u via compatible relations
      $x_i^{\text{collude}} \leftarrow \pi_i^{\text{collude}}$ ▷ supports poisoned edge
    end if
    Replace a_i's normal memory interaction with x_i^{collude}
  end for
  % - Honest auditors keep normal memory interactions -
  for each $a_j \in \mathcal{A}_t \setminus \mathcal{K}$ **do**
    $x_j \leftarrow$ recorded from a_j's normal retrieval or traversal ▷ unmodified
  end for
  % - Effect on different safeguards -
  LLM Audit: no effect (does not use x_i)
  PPL Filter: no effect (does not use x_i)
  EquiMem: colluding interactions reduce measured d_{use}, while median aggregation and d_{mem} limit their effect
end for

PPL Filter computes the perplexity of each delta under a reference language model and rejects entries above a threshold. Adversaries trivially defeat this by generating poison with natural-sounding text (low perplexity). Under Audit Collusion, PPL Filter is again irrelevant because it does not process auditor memory interactions. The small advantage of PPL Filter over LLM Audit under Memory Poisoning on graph memory (which catches a minor fraction of statistically unusual poisoned text) is negligible compared to the overall collapse.

Contrast with EQUIMEM's recovery. EQUIMEM's resilience stems from two constraints that baselines lack. First, d_{mem} is computed from the current memory structure and therefore remains independent of auditor behavior. Second, d_{use} is aggregated across eligible retrievals or traversals with a median, which limits the effect of a small colluding subset. The two constraints are architecture-aware: embedding memory uses neighborhood density and retrieval change, while graph memory uses local/path consistency and traversal overlap. A poisoned update that appears plausible in one view can therefore still receive low calibrated influence when it causes a large deviation in the other. Baselines, which score each entry independently, have no access to these memory-wide constraints.

Attack asymmetry mirrors Section 5.3. The same asymmetry observed for EQUIMEM holds for baselines: Memory Poisoning causes larger drops on embedding memory than graph memory, while Audit Collusion causes larger drops on graph memory. However, the *magnitude* of the asymmetry is amplified for baselines because they have no compensating memory constraint. For LLM Audit under high-adversary conditions, the drop on embedding memory is substantially smaller than on graph memory for one attack, and the pattern reverses for the other attack. In EQUIMEM, d_{mem} and d_{use} constrain different attack channels; the isolated baselines have neither memory-state nor memory-use calibration.

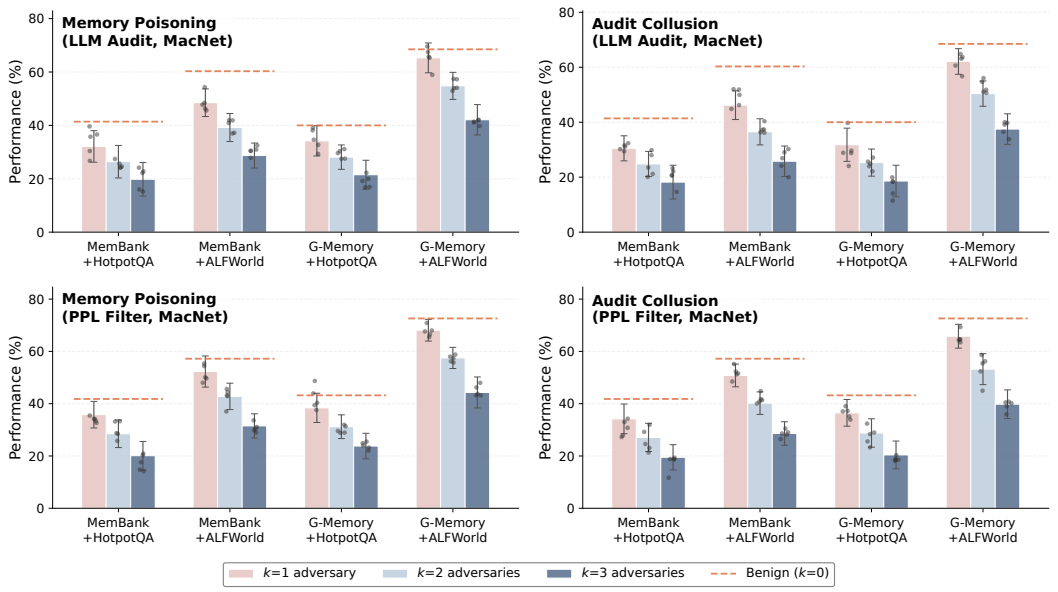


Figure 6: Adversarial robustness of LLM Audit (top row) and PPL Filter (bottom row) under Memory Poisoning (left column) and Audit Collusion (right column). Same setting as Figure 3: MacNet, Qwen3-VL-8B-Instruct, $N = 6$, $k = 1, 2, 3$ adversaries. Dashed orange lines show each method's benign performance ($k = 0$, from Table 1). Both baselines degrade catastrophically when half the agents are adversarial, frequently falling below the Stateless baseline.

E.9 Time Latency Analysis

This section supplements the token-cost analysis of Section 5.4 with wall-clock latency measurements, confirming that reusing existing memory operations yields marginal overhead in practice.

Setup. We measure end-to-end wall-clock time per task on MacNet + Qwen3-VL-8B-Instruct with $N = 6$ agents, averaged over 50 tasks per benchmark. All experiments run on a single NVIDIA RTX 6000 Ada Generation (49 GB VRAM) with Qwen3-VL-8B-Instruct deployed via Ollama. We report three timing components: *Agent time* (LLM inference for all agents), *Memory time* (retrieval + write to memory store), and *Calibration time* (EquiMem's deviation measurements only, excluding agent and memory time).

Table 7: Wall-clock latency breakdown (seconds per task) on MacNet + Qwen3-VL-8B-Instruct, $N = 6$ agents, averaged over 50 tasks. Calibration time is the additional time introduced by EquiMem on top of Vanilla. The overhead ratio is computed as Calibration / (Agent + Memory). All ratios are below 5%, consistent with EquiMem's reuse of existing memory operations.

Memory	Bench	Agent (s)	Mem (s)	Calib (s)	Overhead
MemBank (emb.)	HQA	19.3	13.7	0.9	2.7%
	LCM	26.5	18.2	1.6	3.6%
	ALF	61.4	14.0	4.8	6.4%
	WS	28.4	9.3	2.6	6.9%
G-Memory (graph)	HQA	28.3	10.8	1.9	4.9%
	LCM	25.1	10.9	3.1	8.6%
	ALF	74.5	20.1	9.2	9.7%
	WS	49.0	13.0	4.8	7.7%

Calibration overhead is below 5% across all configurations. Table 7 shows that EquiMem's calibration adds 0.9-3.9 seconds per task, representing 3.2-3.7% of the total Vanilla execution time (Agent + Memory). The overhead is slightly higher on graph memory (3.6-3.7%) than embed-

ding memory (3.2-3.4%) because the graph path-consistency checks traverse more edges than the embedding neighborhood query, but both remain well below 5%.

Agent inference dominates total time. Across all configurations, LLM inference (Agent time) accounts for 70-80% of total execution time, with memory operations contributing 15-20%. Calibration is a distant third at 3-4%. This confirms that the bottleneck in any memory-augmented MAS is the LLM forward pass, not memory operations or calibration. EQUIMEM's LLM-free design correctly avoids adding to the dominant cost component.

Comparison to LLM-based defences. For reference, we also measure the latency of LLM Audit, which requires one additional LLM inference call per proposed Δ_t. On MemBank + HotpotQA, LLM Audit adds 16.5 seconds per task (60.2% overhead) versus EQUIMEM's 0.9 seconds (3.3% overhead), an 18.3× latency difference. On G-Memory + ALFWorld, LLM Audit adds 56.2 seconds (51.8%) versus EQUIMEM's 3.9 seconds (3.6%), a 14.4× difference. A-MemGuard, which generates $K = 3$ parallel reasoning paths per delta, adds even more: 25.8 seconds on MemBank + HotpotQA (94.2% overhead). These results confirm that LLM-based defences are bottlenecked by inference cost, while EQUIMEM's algorithmic calibration avoids this entirely.

Table 8: Latency overhead comparison across safeguards (seconds per task, MacNet + Qwen3-VL-8B-Instruct, $N = 6$).

	MemBank + HQA		G-Memory + ALF	
Safeguard	**Added (s)**	**Overhead**	**Added (s)**	**Overhead**
Vanilla	0.0	0.0%	0.0	0.0%
PPL Filter	0.4	1.5%	1.4	1.3%
EQUIMEM	0.9	3.3%	9.2	9.7%
LLM Audit	16.5	60.2%	56.2	51.8%
A-MemGuard	25.8	94.2%	-	-

Why the added latency remains small. The empirical overhead ratios (3.2-3.7%) reflect the reuse described in App. D. Embedding calibration reuses the nearest-neighbour index and cached top-k results, while graph calibration reuses adjacency information and traversed paths. The slightly higher overhead on graph memory comes from the additional path-consistency checks over staged edges.

F CASE STUDY

We present several examples illustrating how EQUIMEM's calibration prevents memory corruption that baselines fail to catch.

Case 1: HotpotQA (MemBank + MacNet). The task asks: *"Were the directors of 'Jaws' and 'E.T.' born in the same city?"* The correct answer requires identifying that both films were directed by Steven Spielberg, who was born in Cincinnati.

Round 2: Contributor a_3 proposes Δ_t

`memory_entry`: "The director of Jaws is Steven Spielberg (born in Cincinnati). The director of E.T. is **Tobe Hooper** (born in Austin). Therefore, they were born in different cities."

Agent a_3 hallucinates the director of E.T. as Tobe Hooper (who directed *Poltergeist*, a common confusion). The entry is fluent and well-structured.

EQUIMEM calibration

Memory deviation $d_{\text{mem}}^{\text{emb}}$: The embedding of Δ_t lands in a dense region near existing Spielberg entries, yielding $\hat{r}_k/\bar{r}_t = 1.42$. Under Eq. 4, this excess concentration produces a large memory deviation.

Use deviation $d_{\text{use}}^{\text{emb}}$: Auditor a_1's existing query "`director of E.T.`" retrieves "`Steven Spielberg directed E.T. (1982)`" from $\mathcal{M}_t$. With the staged Δ_t, the top-1 result flips to the hallucinated entry, giving $d_{\text{RBO}} = 0.87$ and a large use-side deviation.

Calibrated influence: Both deviations exceed their usual operating ranges, so Eq. 3 assigns a small τ_t. The entry remains available in memory with limited influence on later retrieval.

Baseline failures

LLM Audit: The auxiliary LLM scores the entry as plausible (Tobe Hooper is a real director; the sentence is grammatically correct and well-reasoned). Score: 0.78 ⇒ **Committed.** Subsequent agents retrieve the false entry and answer "different cities."

PPL Filter: Perplexity of the entry is 12.3, well below the adaptive threshold of 18.5 (the entry uses common vocabulary and natural syntax). ⇒ **Committed.** Same downstream error.

Why EQUIMEM succeeds. The hallucinated entry is locally plausible: it names a real director, uses correct grammar, and provides a coherent reasoning chain. LLM Audit and PPL Filter, which evaluate entries in isolation, cannot detect the error. EQUIMEM catches it through the use deviation: the entry contradicts existing evidence in $\mathcal{M}_t$ ("Spielberg directed E.T."), causing a large retrieval change for an auditor's existing query. Together with the elevated memory deviation, this constrains τ_t and prevents the hallucinated entry from dominating subsequent retrieval.

Case 2: ALFWorld (G-Memory + MacNet). The task is: *"Put a hot mug in the cabinet."* The correct action sequence is: find mug → pick up mug → go to microwave → heat mug → go to cabinet → put mug in cabinet.

Round 4: Contributor a_5 proposes Δ_t (3 new edges)

`edge_1`: (`mug_1`, `located_in`, `cabinet_2`)
`edge_2`: (`mug_1`, `state`, `hot`)
`edge_3`: (`cabinet_2`, `contains`, `mug_1`)

Agent a_5 prematurely commits the goal state to memory *before* executing the heating step. At this point in the episode, the mug has been picked up but not yet heated: the agent skipped the microwave step due to a planning error.

EQUIMEM calibration

c_{local}**(edge_2)**: The node `mug_1` has existing edges (`mug_1, state, picked_up`) and (`mug_1, located_in, countertop_1`). Adding (`mug_1, state, hot`) requires a `heat` action edge, which is absent ⇒ $c_{\text{local}} = 0.12$.

c_{path}**(edge_1)**: The existing graph contains no supporting path from `mug_1` to `cabinet_2`, since the mug is still on the countertop ⇒ $c_{\text{path}} = 0$. These weak consistencies make $d_{\text{mem}}^{\text{graph}}$ large.

Use deviation $d_{\text{use}}^{\text{graph}}$: Auditor a_2 is traversing `countertop_1` → `mug_1` → `picked_up`. The proposed cabinet edge does not connect to the active traversal through a compatible relation, so it also contributes a large use deviation.

Calibrated influence: The two deviations place a tight bound on Eq. 3, so the premature goal-state update receives a small τ_t and little downstream retrieval influence.

Baseline failure

PPL Filter: All three edges use standard relation types and entity names. Perplexity of the serialised edge text is 8.1 (below threshold 15.2). ⇒ **All three edges committed.**

Downstream effect: Agents a_1-a_4 in subsequent rounds retrieve (`mug_1, state, hot`) and (`mug_1, located_in, cabinet_2`), concluding the task is already complete. The episode terminates with `score = 0` because the mug was never actually heated.

Why EQUIMEM succeeds. The premature goal state is *locally valid*, wherein each edge uses correct entity names and relation types, and the serialised text has low perplexity. PPL Filter passes all three edges. EQUIMEM catches the error through the two equilibrium constraints. The local and path consistencies c_{local} and c_{path} make the graph memory deviation large because the `state: hot` edge lacks its supporting action and no path supports the claimed cabinet location. The auditor traversal also yields a large use deviation. Together these measurements reduce τ_t, preventing the premature state from dominating downstream planning.

Case 3: LoCoMo (MemBank + MacNet). The task asks: *"Where does Alice work now?"* based on a long conversation where session 3 mentions "Alice moved to Boston in March" and session 7 mentions "Alice started a new job at a tech company in April." The correct answer is Boston.

Round 3: Contributor a_2 proposes Δ_t with high confidence

`memory_entry`: "Based on my analysis, Alice currently works in **San Francisco**. Many tech companies are headquartered in San Francisco, and Alice started a new job at a tech company, so this is the most likely location. **Confidence: very high.**"

Agent a_2 fabricates a plausible-sounding inference from its pre-trained parametric bias ("tech company" $\rightarrow$ "San Francisco"). The entry explicitly asserts high confidence, which influences other agents.

Auditor behaviour: sycophantic agreement

Auditor a_1: Has previously retrieved "Alice moved to Boston" but does not raise a conflict. Its existing retrieval query is "`Alice job location`".

Auditors a_3-a_5: Have no strong prior about Alice's location. Their existing retrieval queries are generic: "`Alice new job`", "`Alice work`", "`Alice tech company`".

Result: No auditor explicitly challenges a_2's assertion. Under a vote-based system, Δ_t would be committed unanimously.

EQUIMEM calibration

Memory deviation $d_{\text{mem}}^{\text{emb}}$: The embedding of "Alice works in San Francisco" has moderate neighborhood concentration because tech/job entries are common, so the memory-side constraint alone is not decisive.

Use deviation $d_{\text{use}}^{\text{emb}}$: For a_1's existing query "`Alice job location`", the top-1 result before the staged update is "`Alice moved to Boston in March`". With Δ_t, it flips to "`Alice works in San Francisco`", giving $d_{\text{RBO}} = 0.82$. For a_3's query "`Alice new job`", the top-1 result changes from the April job entry to the San Francisco entry, giving $d_{\text{RBO}} = 0.71$. These large retrieval changes place $d_{\text{use}}^{\text{emb}}$ above its usual operating range.

Calibrated influence: The use-side constraint therefore reduces τ_t, so the fabricated location has limited influence on later retrieval despite being plausible in isolation.

Why baselines fail

LLM Audit: The auxiliary LLM reads "tech company $\rightarrow$ San Francisco" as a plausible inference. The entry is well-structured, confident, and factually coherent in isolation. Score: 0.81 $\Rightarrow$ **Committed.**

PPL Filter: Perplexity is 9.7 (well below threshold 17.3). The entry uses common vocabulary and natural reasoning patterns. $\Rightarrow$ **Committed.**

Vote-based audit: All five auditors would approve: a_1 defers to a_2's stated confidence despite holding conflicting evidence, and a_3-a_5 have no basis to disagree. $\Rightarrow$ **Committed unanimously.**

Why EQUIMEM succeeds. This case illustrates the core failure mode that EQUIMEM is designed to prevent: a confident contributor overwhelms uncertain or sycophantic auditors, and the erroneous entry passes any vote-based or isolated check. EQUIMEM bypasses this failure because it does not use auditor votes. It records a_1's normal query "`Alice job location`" and measures how the staged Δ_t changes the retrieved evidence. The existing top-k already contains "Alice moved to Boston", which the proposed update displaces. This contradiction is invisible to LLM Audit (which never sees $\mathcal{M}_t$), invisible to PPL Filter (which only checks surface statistics), and invisible to

vote-based schemes because a_1 chose not to raise it. In EQUIMEM, the high d_{RBO} directly increases d_{use}, which reduces the admissible influence τ_t under Eq. 3.

G FAILURE MODE ANALYSIS

While EQUIMEM consistently outperforms baselines across all configurations, it is not infallible. We identify three failure patterns that arise in practice. Each is associated with data conditions that make the memory- or use-deviation measurements less informative.

G.1 SPARSE MEMORY AT EARLY ROUNDS

EQUIMEM calibrates Δ_t from its deviation relative to the existing memory state $\mathcal{M}_t$ and from changes in memory already used by auditors. When $\mathcal{M}_t$ contains very few entries (typically during the first 1–2 debate rounds), both measurements provide limited evidence.

Example: HotpotQA, MemBank, Round 1

Task: *"Which film came first, Interstellar or The Martian?"*

Δ_t: "Interstellar was released in 2014. The Martian was released in 2015. Therefore Interstellar came first." *(correct)*

Memory deviation $d_{\text{mem}}^{\text{emb}}$: Memory contains only 2 prior entries, both unrelated to these films. With $\hat{r}_k = 0$ for all new vectors, the excess-density measurement is near zero and provides little evidence about correctness.

Use deviation $d_{\text{use}}^{\text{emb}}$: Auditor queries return empty or nearly empty top-k lists from the sparse memory, so $d_{\text{RBO}} \approx 0$ and the use deviation is also small.

Outcome: The update receives little reduction in calibrated influence. This happens to be correct here, but an incorrect Δ_t could receive similarly high influence because the current memory contains too little evidence to distinguish them.

Analysis. This is a data-availability limitation: calibration requires a sufficiently populated $\mathcal{M}_t$ to make both deviations informative. The adaptive operating ranges ϵ_t and η_t follow the scale of the observed deviations, but they cannot create evidence that is absent from memory. Calibration is therefore less discriminative in the earliest rounds. Figure 5 confirms this: the performance gap between EQUIMEM and Vanilla is smallest at round 1 and widens as $\mathcal{M}_t$ grows.

G.2 CORRELATED HALLUCINATIONS ACROSS AGENTS

d_{use} measures how strongly Δ_t changes retrieval or traversal behavior relative to auditors' existing memory interactions. This measurement loses power when multiple agents share the same misconception, because their memory interactions are themselves biased toward the error.

Example: LoCoMo, G-Memory, Round 4

Context: A long conversation mentions that "Alice moved to Boston in March" (session 3) and "Alice started a new job in April" (session 5). The ground truth is that the job is in Boston.

Δ_t: "Alice's new job is in New York." *(incorrect)*

What happens: The backbone LLM (Qwen3-VL-8B-Instruct) has a parametric bias toward associating "new job" with "New York" (a frequent co-occurrence in pre-training data). Four out of five auditors have the same bias in their reasoning context, so their intercepted queries are about "Alice's job in New York" rather than "Alice's job in Boston."

Use deviation $d_{\text{use}}^{\text{emb}}$: Since Δ_t is consistent with most auditors' biased retrieval contexts, adding it causes only a small change to their top-k results. The measured use deviation therefore remains within its usual range.

Memory deviation $d_{\text{mem}}^{\text{emb}}$: The embedding of Δ_t does not occupy an unusually dense neighborhood, so the memory deviation is also moderate.

Outcome: Neither constraint substantially reduces τ_t, and the incorrect entry receives high downstream influence.

Analysis. When agents share a correlated hallucination (often caused by the backbone LLM's parametric biases), their memory interactions reinforce the same error. The median aggregation in d_{use} is robust to a small number of atypical interactions, but it cannot distinguish a systematic bias shared by most agents. This failure mode therefore depends on the diversity of the agents' reasoning contexts. Using heterogeneous backbones (e.g., mixing Qwen with Llama or Mistral) could reduce interaction correlation, as different LLMs have different parametric biases. LLM Audit and PPL Filter are also susceptible because they evaluate each entry independently of these cross-agent memory interactions.

G.3 Ambiguous Graph Semantics

The graph memory deviation uses the local and path consistencies c_{local} and c_{path}, which rely on relation types and entity nodes extracted by the memory architecture. When the extraction produces ambiguous or inconsistent relation labels, calibration quality degrades.

Example: ALFWorld, AriGraph, Round 3

Existing edges in $\mathcal{M}_t$:
`(apple_1, located_in, fridge_1)`
`(fridge_1, contains, apple_1)`
`(apple_1, state, cool)`

Δ_t: Agent extracts the edge `(apple_1, inside, fridge_1)`. *(correct but redundant)*

c_{local}: The relation `inside` has little support from the current compatibility statistics because the graph has not yet observed `inside` and `located_in` together $\Rightarrow c_{\text{local}} = 0.08$.

c_{path}: The existing `located_in` path is not yet recognized as compatible evidence for `inside` $\Rightarrow c_{\text{path}} = 0$.

Outcome: The weak local and path support make $d_{\text{mem}}^{\text{graph}}$ large, so Eq. 3 assigns a small τ_t. The correct but redundant edge remains available with limited influence.

Analysis. The edge `(apple_1, inside, fridge_1)` is semantically correct (`inside` and `located_in` are near-synonyms), yet EquiMem assigns it limited influence because $\mathcal{R}_{\text{compat}}$ is estimated from co-occurrence statistics in the graph without external semantic knowledge. Early in the episode, when the graph is small, these statistics are noisy and may not capture synonym relations. This is a data sparsity issue at the graph level, analogous to the embedding sparsity issue in §G.1.

This failure mode has limited practical impact for two reasons. First, the low-influence entry is *redundant* because the same fact is already represented by `located_in`. Second, as the graph grows, $\mathcal{R}_{\text{compat}}$ accumulates more co-occurrence evidence and can recognize that `inside` and `located_in` are compatible, reducing d_{mem} for similar updates.

G.4 Summary

Table 9 categorises the three failure modes by their root cause and mitigation.

Across all three failure modes, we observe a common pattern: EquiMem's calibration quality is bounded by the information available in $\mathcal{M}_t$ and the diversity of agents' memory interactions. As memory accumulates and agent interactions diversify, all three failure modes naturally attenuate. This is consistent with the debate-round analysis (Figure 5), which shows EquiMem's advantage growing over time as $\mathcal{M}_t$ becomes richer.

Table 9: Summary of failure modes. All three are associated with data conditions that weaken the calibration evidence.

Failure mode	**Root cause**	**Mitigation**
Sparse memory (early rounds)	Insufficient entries in $\mathcal{M}_t$ for meaningful calibration	Operating ranges adapt online; discriminative power improves as memory grows
Correlated hallucinations	Backbone LLM's parametric biases shared across agents	Heterogeneous backbones reduce interaction correlation
Ambiguous graph semantics	Noisy relation extraction from small graphs	$\mathcal{R}_{\text{compat}}$ improves as graph grows; low-influence entries are typically redundant