MAKE: DISTRIBUTIONALLY ROBUST KNOWLEDGE EDITING UNDER MULTI-AGENT CONTEXT SHIFT

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge editing for large language models is usually developed and evaluated in isolated settings. In multi-agent systems, role instructions and peer messages change the retrieval context, and thus edits that succeed on clean prompts can fail during interaction. Editing every agent is one fix, but its cost grows with the number of roles. We instead study a setting in which only one designated agent is edited, and ask whether this single intervention can still improve system-level behavior. We identify the main failure mode as interaction-induced representation shift: multi-agent context changes the internal representation used to retrieve the edited fact and can weaken a standard edit. We propose MAKE (**M**ulti-**A**gent **K**nowledge **E**diting), a two-stage framework for multi-agent knowledge editing. *Adversarial Context Mining* collects deployment-relevant contexts, and *Wasserstein-robust multi-key editing* enforces the target association on clean and contextual keys while minimizing a tractable upper bound on worst-case off-support activation. We also derive an off-support sensitivity bound for unseen contextual deviations. Experiments on two LLM families and three representative multi-agent architectures show large drops from isolated edit success to final system success for standard editors. MAKE improves multi-agent edit success and conflict robustness while maintaining competitive generalization and preservation under comparable computational cost. It also remains stable under sequential edits, achieving limited degradation of general-purpose capabilities.[1]

1 INTRODUCTION

Large language models (LLMs) are now widely used in multi-agent systems (MAS), where role-specialized agents exchange intermediate results and solve tasks jointly (Wu et al., 2024; Hong et al., 2024; Qian et al., 2025; Yang et al., 2026). These systems inherit the factual weaknesses of their underlying models. Parametric knowledge becomes outdated (Liu et al., 2026; Tang et al., 2026), and generated responses may contain hallucinated claims (Weng et al., 2025; Jamshidi, 2026; Li et al., 2026). Knowledge editing updates specific facts without retraining the full model (Yao et al., 2023; Cohen et al., 2024). Existing editors modify a small set of parameters, rewrite internal key–value associations, or attach auxiliary memories (Meng et al., 2022; 2023; Fang et al., 2025; Ikram et al., 2026), but they are usually designed and evaluated for a single model under a clean prompt. This gap shows up in multi-agent deployment.

One option is to edit every agent. Figure 1 shows that this raises editing and maintenance cost with the number of roles, and can also hurt system accuracy and capability preservation. We therefore study a cheaper setting in which only one designated agent is edited. The question is whether a single edit can still produce reliable system-level corrections.

The difficulty is that interaction changes the retrieval conditions of the edited fact. The edited agent sees not only the original query, but also role instructions and peer messages. These inputs can shift the internal representation used to retrieve the fact, so an edit that works on a clean prompt may weaken or fail in interaction. We study this effect at two levels. At the representation level, contextual shifts change the response of the edited layer, and low-rank updates may miss important shift directions. At the system level, a local retrieval failure can propagate through later communication.

[1] Source code and data are provided in the supplementary material and will be released upon acceptance.

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.

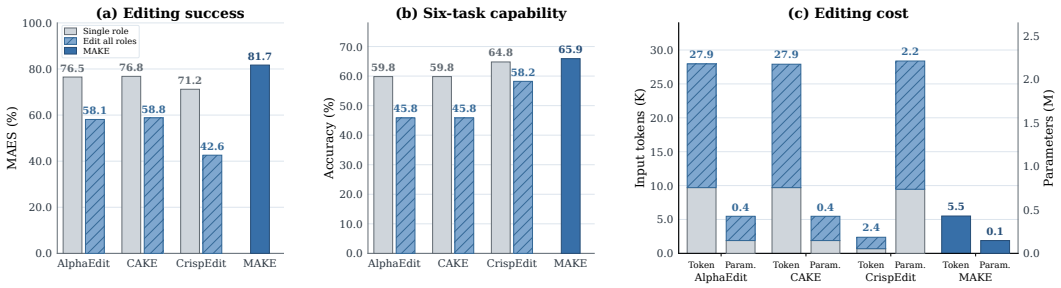

Figure 1: Comparison of performance and computational cost between MAKE and typical single-agent knowledge editing methods in MAS scenario.

This points to an editing procedure that covers the interaction contexts the edited agent is likely to encounter at deployment.

We propose MAKE (Multi-Agent Knowledge Editing), a framework for robust editing under multi-agent deployment. It has two stages. *Adversarial Context Mining* runs upstream agents and collects interaction contexts that induce large representation shifts for the edited role, while keeping the deterministic evaluation trajectory as an anchor. *Wasserstein-robust multi-key editing* then enforces the new fact on the clean prompt and mined contextual keys while minimizing a tractable Wasserstein upper bound on off-support activation (Esfahani & Kuhn, 2018). The resulting update has a closed form, and our sensitivity analysis links its robustness to unseen contextual deviations.

We evaluate MAKE on Qwen2.5-7B (Yang et al., 2025) and Llama-3.1-8B (Grattafiori et al., 2024) across three representative multi-agent structures (Tran et al., 2025). Existing editing methods achieve a low overall average score of **30.04%** across all benchmarks in final team responses. MAKE improves by **+11.6%** over the strongest baseline, while balancing editing success, conflict robustness, generalization, and preservation. After 200 sequential edits, MAKE's average Macro-F1 across general-capability tasks decreases by only **-0.3%**. Beyond superior performance stability, our method also exhibits comparable low inference computational overhead. These results show that robust multi-agent editing does not require modifying every agent.

2 Problem Setup and Representation Shift

We study knowledge editing in a MAS under a minimal intervention: one designated agent is edited while all other agents retain their original parameters. We analyze how interaction-induced key shifts weaken an edit that succeeds on a clean prompt. All proofs are deferred to Appendix C.

2.1 From Isolated to Multi-Agent Editing

A factual edit is $e = (s, r, o \to o^*)$, where o^* replaces the original answer o, and q_e is its clean query. A MAS is $\mathcal{M} = (\mathcal{A}, \mathcal{G}, \Pi)$, where $\mathcal{A} = \{a_i = (f_{\theta_i}, p_i)\}_{i=1}^n$ contains role-conditioned agents, $\mathcal{G}$ is the communication graph, and Π is the message schedule. At step t, agent i_t generates $m_t \sim f_{\theta_{i_t}}(\cdot \mid p_{i_t}, m_{<t}, q_e)$. For an intervention set $\mathcal{S}$, define $\theta_i^{\text{edit}} = \theta_i + \mathbb{1}[a_i \in \mathcal{S}]\Delta\theta_i$ and $C_{\text{edit}}(\mathcal{S}) = \sum_{a_i \in \mathcal{S}} c_i$. The general deployment objective is

$$\min_{\mathcal{S}, \{\Delta\theta_i\}_{a_i \in \mathcal{S}}} \quad C_{\text{edit}}(\mathcal{S}) \tag{1}$$

$$\text{s.t.} \quad \mathbb{E}\big[\phi_e\big(\mathcal{M}_{\mathcal{S}}^{\text{edit}}(q_e)\big)\big] \geq \tau, \quad \frac{1}{|\mathcal{T}_e^{\text{pres}}|} \sum_{(q,y) \in \mathcal{T}_e^{\text{pres}}} \mathbb{E}\big[\phi_{q,y}\big(\mathcal{M}_{\mathcal{S}}^{\text{edit}}(q)\big)\big] \geq \tau_{\text{pres}}, \tag{2}$$

where $\phi_{q,y}$ tests whether a response expresses reference y. We fix the task-relevant agent $a_e = (f_{\theta_e}, p_e)$ and set $\mathcal{S} = \{a_e\}$; optimal role selection is outside our scope.

The clean-to-agent gap measures disruption from interaction context, while the agent-to-system gap measures subsequent propagation and aggregation. Hence, we distinguish $E_{\text{clean}} = \{\phi_e(f_{\theta_e + \Delta\theta_e}(q_e)) = 1\}$, $E_{\text{system}} = \left\{\phi_e\left(\mathcal{M}_{\{a_e\}}^{\text{edit}}(q_e)\right) = 1\right\}$, $E_{\text{agent}}(m_{<t_e}) = \{\phi_e(f_{\theta_e + \Delta\theta_e}(p_e, m_{<t_e}, q_e)) = 1\}$, where t_e is the turn at which a_e responds.

2.2 Representation Shift and Attenuation

At layer l, let $h_l(x) \in \mathbb{R}^d$ be the subject-token hidden state and define the MLP key

$$k_l(x) = \sigma(W_l^{\text{in}} h_l(x)), \qquad \text{MLP}_l(h_l(x)) = W_l^{\text{out}} k_l(x),$$

where $k_l(x) \in \mathbb{R}^{d_m}$. Fixing l, write $W = W_l^{\text{out}}$ and $k = k_l(q_e)$. For context $c = (p_e, m_{<t_e})$, let $x_c = (p_e, m_{<t_e}, q_e)$, $k_c = k_l(x_c)$, and $\delta(c) = k_c - k$. Let v^* be the clean target value and let $C \succ 0$ be a regularized estimate of the background-key second moment. The clean-key rank-one update is

$$\Delta W = (v^* - Wk)\frac{k^\top C^{-1}}{k^\top C^{-1} k}, \quad \widehat{W} = W + \Delta W, \tag{3}$$

which satisfies $\widehat{W}k = v^*$.

Theorem 2.1 (Contextual edit decomposition). *Define* $\eta(c) = \frac{k^\top C^{-1} k_c}{k^\top C^{-1} k}$. *Then*

$$\Delta W k_c = (v^* - Wk)\eta(c), \quad \widehat{W}k_c - v^* = \underbrace{W\delta(c)}_{\textit{backbone shift}} + \underbrace{(v^* - Wk)(\eta(c) - 1)}_{\textit{edit attenuation}}. \tag{4}$$

Thus, interaction affects contextual retrieval through both the base model's response to the shifted key and the rescaling of the edit itself.

2.3 Blind Directions and Propagation

Let a rank-r update have compact SVD $\Delta W = U_r S_r V_r^\top$, and define $P_r = V_r V_r^\top$, $P_r^\perp = I - P_r$, $\delta_\parallel = P_r \delta$, and $\delta_\perp = P_r^\perp \delta$.

Theorem 2.2 (Uncovered shift directions). *If* $\widehat{W}k = v^*$, *then*

$$\Delta W \delta_\perp(c) = 0, \quad \widehat{W}k_c - v^* = W\delta_\perp(c) + \widehat{W}\delta_\parallel(c). \tag{5}$$

A low-rank update is therefore insensitive to shift components outside its key-domain row space, although the base mapping still responds to them.

This local error can propagate through later agents. Consider a path of L stages from the edited agent to the final response. Let $G_1 = E_{\text{agent}}(m_{<t_e})$, let G_t denote correctness after stage t, and define as $F_t = \Pr\left(G_t \,\middle|\, \bigcap_{j=1}^{t-1} G_j\right), \quad t = 2, \ldots, L.$

Proposition 2.3 (Pathwise propagation). *If failure is absorbing and* $G_L = E_{\text{system}}$, *then*

$$\Pr(E_{\text{system}}) = \Pr(E_{\text{agent}}) \prod_{t=2}^{L} F_t.$$

If later stages may repair errors but correctness along the entire path is sufficient for final success, the equality becomes a lower bound.

This result requires no independence assumption and shows how a contextual retrieval failure can accumulate along the communication path.

2.4 Compact Interaction Shifts

Although $\delta(c) \in \mathbb{R}^{d_m}$ is high-dimensional, interaction shifts may occupy a compact subspace. For contexts $c \sim \mathcal{P}_c$, define

$$\mu_\delta = \mathbb{E}[\delta(c)], \qquad \Sigma_\delta = \mathbb{E}\left[(\delta(c) - \mu_\delta)(\delta(c) - \mu_\delta)^\top\right].$$

Let $\lambda_1, \ldots, \lambda_{q_\delta}$ be the positive eigenvalues of Σ_δ. Its effective dimension is

$$d_{\text{eff}} = \begin{cases} \dfrac{\left(\sum_{j=1}^{q_\delta} \lambda_j\right)^2}{\sum_{j=1}^{q_\delta} \lambda_j^2} = \dfrac{[\text{Tr}(\Sigma_\delta)]^2}{\text{Tr}(\Sigma_\delta^2)}, & q_\delta > 0, \\ 0, & q_\delta = 0. \end{cases} \tag{6}$$

Theorem 2.4 (Low-dimensional interaction shift). *Suppose* $h_l(x_c) - h_l(q_e) = B_{\text{ctx}}\zeta(c) + b_{\text{ctx}}$, *where* $\zeta(c) \in \mathbb{R}^{r_c}$ *has finite second moment. The local key map is affine with linear part* J_l. *Then*

$$\delta(c) = J_l(B_{\text{ctx}}\zeta(c) + b_{\text{ctx}}), \qquad \text{rank}(\Sigma_\delta) \le r_c, \qquad d_{\text{eff}} \le r_c. \tag{7}$$

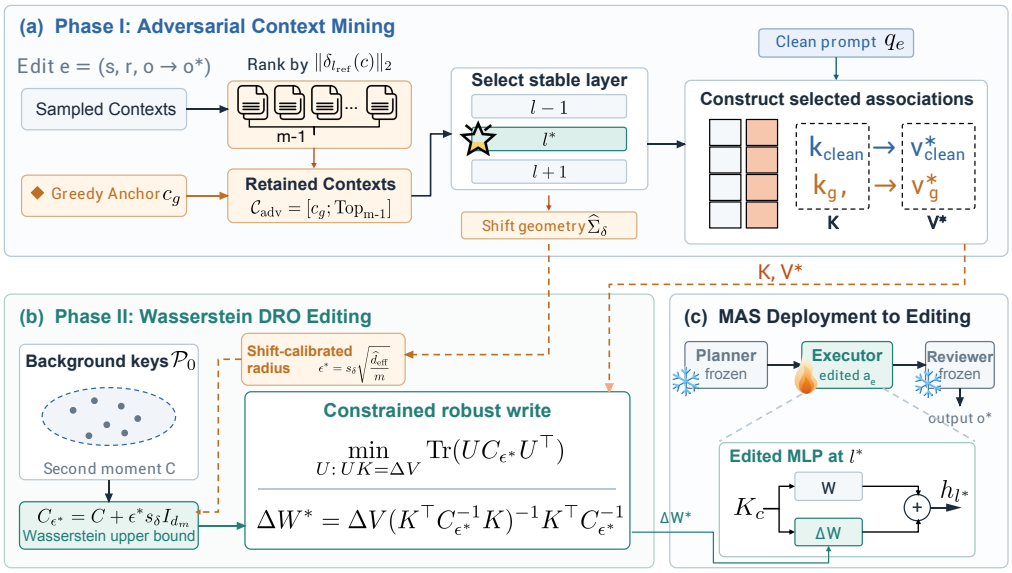

Figure 2: Overview of MAKE. Phase I mines multi-agent contexts that expose the edit. Phase II performs a robust write over the resulting clean and contextual key–value associations.

3 METHOD

Given a task-relevant agent a_e, MAKE operates at the minimal intervention point $\mathcal{S} = \{a_e\}$. It has two phases. Phase I mines interaction contexts that induce large representation shifts for the edited role and selects an edit layer. Phase II performs a shift-calibrated multi-key write over the resulting clean and contextual associations. Figure 2 summarizes the framework.

3.1 PHASE I: ADVERSARIAL CONTEXT MINING

Given an edit assigned to a_e, we run the upstream agents under graph $\mathcal{G}$ and schedule Π to produce a candidate set $\mathcal{C}_{\text{pool}}$ of contexts $c = (p_e, m_{<t_e})$. Phase I returns $(\mathcal{C}_{\text{adv}}, l^*, \widehat{\Sigma}_\delta)$.

At layer l, each context induces a key shift $\delta_l(c) = k_l(x_c) - k_l(q_e)$, where x_c is the context-conditioned input from Section 2.2. We obtain a reference layer l_{ref} using the ROME-style causal-tracing procedure for factual recall (Meng et al., 2022)[2]. We rank candidate contexts by $d(c) = \|\delta_{l_{\text{ref}}}(c)\|_2$, which measures how far interaction moves the edited representation from its clean-prompt key. Let c_g be the greedy trajectory used as the deterministic deployment anchor in the main evaluation. For a budget m, we keep

$$\mathcal{C}_{\text{adv}} = \left[c_g;\ \text{Top}_{m-1}\left(\mathcal{C}_{\text{pool}} \setminus \{c_g\}; d\right)\right]. \tag{8}$$

The greedy anchor matches the main deterministic evaluation, and the remaining contexts cover high-shift interaction traces. Section B.6 evaluates the frozen edit on independently generated peer-message trajectories beyond this anchor.

Conditioned on the retained contexts, we choose the edit layer over $\mathcal{L}$[3]:

$$l^* = \arg\min_{l \in \mathcal{L}} \frac{1}{|\mathcal{C}_{\text{adv}}|} \sum_{c \in \mathcal{C}_{\text{adv}}} \left[1 - \cos\left(k_l(q_e), k_l(x_c)\right)\right]. \tag{9}$$

This favors layers where clean and contextual keys remain aligned, which stabilizes the joint write. For $\delta_j = \delta_{l^*}(c_j)$ with $c_j \in \mathcal{C}_{\text{adv}}$ and $m = |\mathcal{C}_{\text{adv}}|$, let $\widehat{\mu}_\delta = \frac{1}{m}\sum_{j=1}^m \delta_j$. The empirical shift

[2] Its architecture-specific configuration is given in Appendix A.4.5.

[3] A predetermined architecture-specific set of editable MLP layers, fixed before context mining and specified in Appendix A.4.5.

covariance is

$$\widehat{\Sigma}_\delta = \frac{1}{m}\sum_{j=1}^{m}(\delta_j - \widehat{\mu}_\delta)(\delta_j - \widehat{\mu}_\delta)^\top. \tag{10}$$

3.2 PHASE II: WASSERSTEIN DRO EDITING

Phase II turns the output of Phase I into a single constrained update. Let $W = W_{l^*}^{\text{out}}$. For each retained context $c_j = (p_e, m_{<t_e}^{(j)})$, $j = 1, \ldots, m$, define the corresponding input $x_{c_j} = (p_e, m_{<t_e}^{(j)}, q_e)$. Index the clean key by 0 and the contextual keys by $1, \ldots, m$: $k^{(0)} = k_{l^*}(q_e)$ and $k^{(j)} = k_{l^*}(x_{c_j})$. We stack these keys and target values as: $K = \left[k^{(0)}, k^{(1)}, \ldots, k^{(m)}\right] \in \mathbb{R}^{d_m \times (m+1)}$ and $V^* = \left[v^*, v_g^*, \ldots, v_g^*\right] \in \mathbb{R}^{d \times (m+1)}$, $\Delta V = V^* - WK$. Here v^* is the clean target value in Sec. 2.2, and v_g^* is the contextual target value optimized on the greedy anchor. The first columns of K and V^* define the clean association; the remaining m columns pair each contextual key with v_g^*.

For an update $U \in \mathbb{R}^{d \times d_m}$, exact coverage of the selected associations requires

$$(W + U)K = V^* \quad \Longleftrightarrow \quad UK = \Delta V. \tag{11}$$

This fixes the update on the selected keys but not elsewhere, so we regularize it over nearby background key distributions. Let $\mathcal{P}_0$ be the nominal distribution of background keys $z \in \mathbb{R}^{d_m}$ at layer l^*, and let $M_0 = \mathbb{E}_{z\sim\mathcal{P}_0}[zz^\top]$ be its second moment. As in Section 2.2, $C \succ 0$ is a positive-definite regularized estimate of this moment, and for the upper bound below we assume $C \succeq M_0$. For a radius $\epsilon \geq 0$, define

$$\mathcal{R}_\epsilon(U) = \sup_{\mathcal{Q}\in\mathbb{B}_\epsilon(\mathcal{P}_0)} \mathbb{E}_{z\sim\mathcal{Q}}\|Uz\|_2^2, \tag{12}$$

where $\mathcal{Q}$ ranges over probability distributions on $\mathbb{R}^{d_m}$ with finite second moment, $\mathcal{W}_2$ is the 2-Wasserstein distance, and $\mathbb{B}_\epsilon(\mathcal{P}_0) = \{\mathcal{Q} : \mathcal{W}_2(\mathcal{Q}, \mathcal{P}_0) \leq \epsilon\}$. Equation eq. (11) enforces the selected clean and contextual constraints, while eq. (12) controls the update over nearby key distributions.

Proposition 3.1 (Tractable Wasserstein upper bound)**.** *If $\mathcal{P}_0$ has finite second moment and $C \succeq M_0$, then for every $\beta > 0$,*

$$\mathcal{R}_\epsilon(U) \leq (1+\beta)\,\mathrm{Tr}(UCU^\top) + (1+\beta^{-1})\epsilon^2\|U\|_F^2 = (1+\beta)\,\mathrm{Tr}\left[U\left(C + \frac{\epsilon^2}{\beta}I_{d_m}\right)U^\top\right]. \tag{13}$$

The proof is given in Appendix C.5. Let $\widehat{d}_{\text{eff}}$ denote the effective dimension in eq. (6) evaluated with $\widehat{\Sigma}_\delta$ in place of Σ_δ, with the same zero-covariance convention. We use the empirical radius

$$s_\delta = \sqrt{\frac{\mathrm{Tr}(\widehat{\Sigma}_\delta)}{d_m}}, \quad \epsilon^* = s_\delta\sqrt{\frac{\widehat{d}_{\text{eff}}}{m}}, \quad C_{\epsilon^*} = C + \epsilon^* s_\delta I_{d_m}. \tag{14}$$

When $s_\delta > 0$, choosing $\beta^* = \epsilon^*/s_\delta$ in eq. (13) yields the inflation in C_{ϵ^*}; when $s_\delta = 0$, eq. (14) gives $\epsilon^* = 0$ and $C_{\epsilon^*} = C$. Dropping the positive multiplicative factor in the upper bound, we solve $\Delta W^* = \arg\min_{U:\,UK=\Delta V} \mathrm{Tr}(UC_{\epsilon^*}U^\top)$. If $C_{\epsilon^*} \succ 0$ and $K^\top C_{\epsilon^*}^{-1}K$ is nonsingular, the unique solution is

$$\Delta W^* = \Delta V(K^\top C_{\epsilon^*}^{-1}K)^{-1}K^\top C_{\epsilon^*}^{-1}. \tag{15}$$

Its minimum covariance-weighted cost is

$$\min_{U:\,UK=\Delta V} \mathrm{Tr}(UC_{\epsilon^*}U^\top) = \mathrm{Tr}\left[\Delta V(K^\top C_{\epsilon^*}^{-1}K)^{-1}\Delta V^\top\right]. \tag{16}$$

Corollary 3.2 (Off-support sensitivity of the robust write)**.** *Let $U^* = \Delta W^*$ solve ΔW^*, $\widehat{W} = W + U^*$, and assume $C_{\epsilon^*} \succ 0$. For every key deviation $e \in \mathbb{R}^{d_m}$,*

$$\|U^*e\|_2^2 \leq \left(e^\top C_{\epsilon^*}^{-1}e\right)\mathrm{Tr}\left(U^*C_{\epsilon^*}(U^*)^\top\right). \tag{17}$$

Moreover, consider an unseen contextual association $k' = K\alpha + e_k$ and $v' = V^*\alpha + e_v$, where $\alpha \in \mathbb{R}^{m+1}$, $e_k \in \mathbb{R}^{d_m}$, and $e_v \in \mathbb{R}^d$. Because $\widehat{W}K = V^*$, its value residual satisfies

$$\|\widehat{W}k' - v'\|_2 \leq \|We_k - e_v\|_2 + \sqrt{e_k^\top C_{\epsilon^*}^{-1}e_k}\,\sqrt{\mathrm{Tr}(U^*C_{\epsilon^*}(U^*)^\top)}. \tag{18}$$

Table 1: Main knowledge editing results with Llama 3.1 (8B). Qwen2.5 results and additional architectures are given in Appendix B.1. The metrics S/M/A/G/P denote SAES/MAES/ACR/MAG/MAP explained in Appendix A.2. Bold and underlined numbers mark the best and second-best results.

Method	CounterFact					zsRE					RippleEdits					UniEdit					Avg.
	S	M	A	G	P	S	M	A	G	P	S	M	A	G	P	S	M	A	G	P	
Pre-Edited	1.1	1.5	3.5	1.2	30.8	26.3	23.5	31.6	21.8	22.0	0.2	0.1	0.4	3.4	22.4	9.0	9.4	12.2	2.9	4.8	11.4
FT	19.7	31.3	71.0	21.6	24.4	58.5	59.8	70.0	54.9	19.3	5.9	17.3	22.6	14.5	19.9	45.3	44.5	57.6	8.1	6.1	33.6
MEND	31.0	28.2	66.2	23.9	28.3	49.6	45.1	54.6	41.8	20.8	14.0	17.1	26.0	13.2	24.0	32.9	29.1	36.3	6.4	5.7	29.7
ROME	79.8	10.5	48.6	7.9	29.9	79.2	43.3	65.4	33.8	21.0	37.6	5.8	27.6	4.6	24.0	66.2	25.7	42.4	4.6	5.1	33.2
MEMIT	2.7	30.9	24.1	2.6	30.3	35.3	57.2	47.0	29.0	21.2	0.7	10.5	3.5	4.4	**24.1**	15.4	32.3	21.8	3.7	5.1	20.1
AlphaEdit	2.9	33.5	27.8	3.0	30.3	35.3	61.8	47.2	29.9	20.8	0.6	23.5	4.6	6.4	24.0	16.1	43.2	23.4	3.8	5.3	22.2
CAKE	2.9	33.2	28.7	3.2	**30.4**	35.4	62.4	47.3	29.9	21.2	0.7	23.2	4.8	6.6	**24.1**	16.3	42.9	23.6	3.6	5.1	22.3
CrispEdit	71.0	33.4	75.0	33.0	25.0	61.4	60.9	63.1	**58.3**	18.4	23.7	28.3	60.7	22.1	22.1	51.4	41.2	47.6	**9.0**	6.1	40.6
MAKE	**96.8**	**36.5**	**85.1**	**33.4**	29.4	**95.3**	**65.5**	**80.9**	55.0	**21.3**	**90.9**	**32.2**	**61.5**	**23.0**	**24.1**	**92.8**	**52.1**	**67.1**	7.8	**6.2**	**52.8**

Among all updates that exactly satisfy the selected multi-key constraints, MAKE minimizes the shared energy term that controls the edit-induced response to deviations outside the selected support. The remaining term $\|We_k - e_v\|_2$ captures mismatch already present in the base mapping. The only explicit inverse is the $(m+1) \times (m+1)$ Gram matrix. In the single-key limit, $m = 0$ and $\epsilon^* = 0$, eq. (15) recovers the rank-one update in eq. (3) (Meng et al., 2022). The closed-form derivations, off-support proof, and stabilized implementation are given in Appendix C.6.

3.3 EDIT-TIME OPTIMIZATION

MAKE requires no dataset-level training or editor pretraining. For each edit request, Phase I builds the interaction-conditioned support and estimates its shift geometry, and Phase II computes ΔW^* in closed form after target-value optimization with the base model frozen. At the minimal intervention point $\mathcal{S} = \{a_e\}$, the resulting update is activated only for the designated agent a_e. All other agents keep their original parameters and continue to induce context shifts during interaction. Implementation details are deferred to Appendix A.4.

4 EXPERIMENTS

Due to the space limitation, the datasets, baselines, and implementations are shown in Appendix A.

4.1 MULTI-AGENT EDITING RESULTS

We evaluate MAKE and the baselines on CounterFact, zsRE, RippleEdits, and UniEdit with Llama 3.1 (8B) and Qwen2.5 (7B), editing only the designated role and leaving peer agents unchanged. Table 1 reports the Llama results under the nominal greedy interaction,[4] and Section B.6 evaluates the same frozen edits under independently generated peer messages. Metric definitions are given in Appendix A.2.

Table 1 and Appendix B.1 show three patterns. First, fitting the target fact is not enough for reliable multi-agent behavior. FT improves clearly over the pre-edited model, but its ACR, MAG, and MAP remain uneven, suggesting that target-answer fitting alone does not resolve role conditioning, peer messages, or conflicting evidence. Second, existing editors cover only part of the failure. ROME and several other baselines often achieve strong SAES but much lower MAES, so clean-query success does not carry over to the final system response. Multi-layer and preservation-oriented methods improve editing capacity, generalization, or locality, but remain inconsistent on MAES and ACR because they do not constrain interaction-shifted representations. Third, MAKE performs best at the system level. Its gains on both ACR and MAES indicate stronger resistance to conflicting messages and more reliable survival through downstream aggregation, while MAG and MAP remain competitive. Since an ineffective edit can score well on preservation by failing to learn the target fact, MAP should be interpreted together with editing efficacy. Overall, the results show a clear gap between isolated editing and multi-agent deployment, and MAKE narrows that gap.

[4] Results on Qwen2.5 (7B) are reported in Appendix B.1.

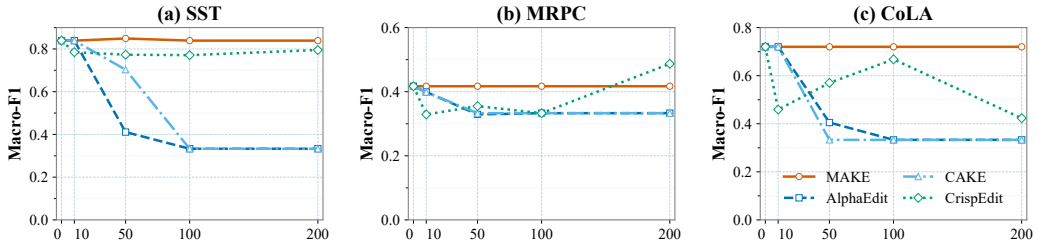

Figure 3: Sequential editing stability and preservation of general capabilities. The other general tasks are shown in Appendix B.1.

4.2 General Ability Evaluation

We test whether repeated edits degrade general capabilities. Following the AlphaEdit protocol (Fang et al., 2025), we apply CounterFact edits sequentially and evaluate the edited model at successive checkpoints. We use the same six general ability tasks, comparing MAKE with AlphaEdit (Fang et al., 2025), CAKE (Liu & Zhang, 2026), and CrispEdit (Ikram et al., 2026).

Figure 3 shows three patterns. First, general capabilities decline as edits accumulate. All methods degrade with more sequential edits, and thus localized updates still create cumulative interference. CrispEdit retains more capability, but does not jointly control interaction-conditioned support and background activation. Second, MAKE slows this degradation. Although its performance also decreases slightly, it retains the strongest overall capability and the flattest curves across all six tasks. This suggests that each edit supports robust factual retrieval while causing less disruption to unrelated representations. Third, this trend holds across diverse capabilities. MAKE remains strongest on language understanding, factual knowledge, and reasoning, so the effect is not limited to a single task type or benchmark. Together with Sec. 4.1, these results show the shift-aware support and distributionally robust control to improve MAS editing while preserving general capabilities.

4.3 Ablation Study

Table 2 reports the ablation results on CounterFact with Llama 3.1 (8B). Replacing adversarial context selection with random selection keeps the clean and greedy keys but samples the other four contexts uniformly from the same candidate pool. SAES changes little, but MAES, ACR, and MAG drop sharply, which shows that the gains of MAKE come from selecting high-shift contexts, not simply from adding more contextual constraints.

Table 2: Ablation study on CounterFact with Llama 3.1 (8B). RS and Adaptive LS denote random selection and adaptive layer selection.

Variant	SAES	MAES	ACR	MAG	MAP	Avg.
MAKE (full)	**97.5**	**35.1**	**90.1**	**27.3**	**30.0**	**56.0**
- RS	97.3	19.3	71.8	22.6	28.4	47.9
- Greedy Only	97.4	12.4	73.6	22.1	28.0	46.7
- DRO	97.1	24.6	75.4	23.3	29.4	50.0
- Adaptive LS	95.3	24.7	87.6	24.4	29.5	52.3

Keeping only the clean and greedy keys reduces system-level performance even further, which suggests that a single nominal trajectory does not cover the shifts induced by diverse peer interactions. Removing DRO keeps the same multi-key support but disables the Wasserstein regularizer. The resulting drop is smaller but consistent, especially on ACR and MAG, which indicates that contextual multi-key fitting accounts for most of the gain, while robust regularization adds protection against conflicting and off-support contexts. Removing adaptive layer selection and editing a fixed layer also lowers performance, though less than the context-selection ablations. This is consistent with the view that layers with better clean-context alignment support a more stable joint write.

4.4 Detailed Analysis

4.4.1 Edit Survival under Multi-Round Interaction

A deployed MAS may exchange and aggregate responses over multiple rounds, so we test whether an edited fact survives longer interaction trajectories. Using 500 CounterFact and 500 zsRE edits, we evaluate Pipeline, Debate, and Hierarchical architectures with rounds in $\{1, 2, 3, 5\}$. The same edit

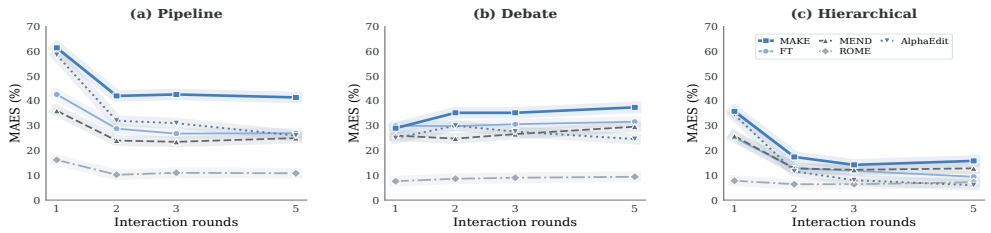

Figure 4: Multi-round editing results on CounterFact. zsRE results are shown in Appendix B.3.

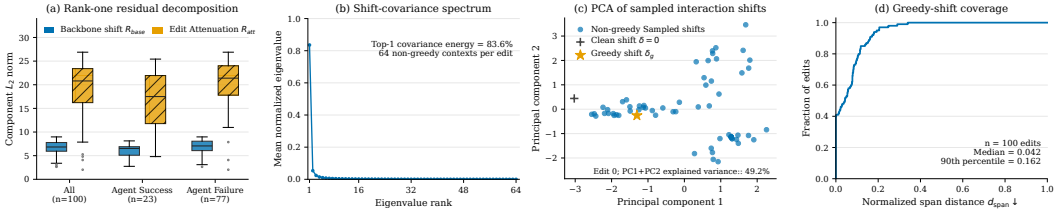

Figure 5: Empirical validation of Representation Shift Theory.

is held fixed across rounds with no additional updates. We compare MAKE with ROME, AlphaEdit, FT, and MEND, and report MAES. Figure 4 shows that MAKE achieves the highest MAES in almost every round across all three architectures. Pipeline and Hierarchical performance gradually declines because repeated processing and aggregation by unedited downstream agents can dilute the edited evidence or reintroduce the pre-edit answer. In contrast, Debate improves slightly before converging, as iterative exchange allows the corrected fact to be repeatedly reinforced and incorporated by the final judge. Across these different dynamics, MAKE consistently outperforms all baselines, showing that Adversarial context mining covers consequential interaction shifts while Wasserstein-robust multi-key editing improves robustness to contexts arising over longer communication trajectories.

4.4.2 Validation of Representation Shift Theory

We directly test three implications of Representation Shift Theory using 100 CounterFact edits with Llama 3.1 (8B). For each edit, we sample 64 interaction contexts for this diagnostic analysis and evaluate the quantities at the rank-one edit layer. Following eq. (4), we measure the magnitudes of the backbone-shift and edit-attenuation terms as $R_{\text{base}}(c) = \|W\delta(c)\|_2, \quad R_{\text{att}}(c) = \|(v^* - Wk)(\eta(c) - 1)\|_2$ and compare their distributions over all, successful, and failed contextual queries. We then compute the normalized eigenspectrum of the centered empirical shift covariance and its plug-in effective dimension $\widehat{d}_{\text{eff}}$ using eq. (6). Finally, let $\delta_g = \delta(c_g)$ denote the greedy shift and P_{samp} be the orthogonal projector onto the span of the non-greedy sampled shifts. For $\|\delta_g\|_2 > 0$, we measure $d_{\text{span}} = \frac{\|(I - P_{\text{samp}})\delta_g\|_2}{\|\delta_g\|_2}$ and visualize the sampled and greedy shifts using PCA.

Figure 5 provides consistent evidence for the three theoretical mechanisms to connect the theory to both components of MAKE. First, the edit-attenuation term is substantially larger than the backbone-shift term and becomes stronger among failed queries, supporting the prediction that interaction can weaken a clean-key edit by changing its effective response through $\eta(c)$. Second, the rapid spectral decay and low empirical effective dimension show that interaction-induced shifts occupy a compact, structured subspace rather than spreading uniformly across the key space. The PCA confirms this structure, while small d_{span} indicates broad greedy-shift coverage and residual components justify retaining an exact anchor.

4.4.3 Computational Cost Analysis

We measure whether the multi-agent robustness of MAKE introduces substantial computational overhead using 100 CounterFact edits with Llama 3.1-8B. We separate one-time edit computation from recurring inference, since a constructed patch can be reused across many queries and interaction trajectories. We vary the number of retained contexts over $m \in \{0, 5, 10, 20\}$ and report both uncached runs and cached runs that reuse previously constructed contexts and patches. Table 3 shows that MAKE achieves substantial performance gains with computational costs comparable to existing editors. At the default $m = 5$, its editing time and peak memory remain in the same range

Table 3: Comparison of editing cost. The context scalability results are shown in Appendix B.4. ♣ and † denote uncached and cached runs. Cached runs reuse both context and patch caches.

Method / setting	Mining (s)	Update (s)	Total edit (s)	Infer. (s)	Infer. Overhead	Peak infer. memory (GB)	Avg. score (%)
FT	13.78	0.553	14.331	3.578	2.02%	22.52	21.2
MEND	13.92	2.286	16.206	3.546	2.04%	30.70	50.9
ROME	13.15	0.453	13.606	3.446	2.08%	19.29	38.4
MEMIT	13.27	2.339	15.608	3.621	14.93%	19.07	20.5
AlphaEdit	13.31	3.091	16.398	3.849	14.67%	18.31	20.7
♣ MAKE, $m = 5$	17.18	1.448	18.625	3.599	2.07%	19.31	63.1

as the baselines, while its 2.07% inference overhead is close to that of lightweight methods. The additional one-time cost mainly comes from context generation, whereas the closed-form update remains efficient as the context budget grows. Moreover, performance improves markedly after introducing interaction contexts, while larger budgets provide only marginal gains, supporting the use of a compact context set. Since each patch can be reused across queries, the edit-time cost is amortized in deployment, improving our MAS robustness without substantial recurring overhead.

4.5 Other Experiments

Due to the space limitation, we present other experimental analyses, including **Role-location sensitivity** in Appendix B.5, **Held-out peer-message evaluation** in Appendix B.6 and **Scaling adversarial strength** in Appendix B.7, respectively.

5 Related Work

Knowledge editing. Knowledge editing modifies a model's response to selected facts while preserving unrelated capabilities. Early methods fine-tune selected parameters or learn hypernetworks that predict updates (De Cao et al., 2021; Mitchell et al., 2022). Locate-then-edit methods treat transformer MLPs as associative memories (Geva et al., 2021): ROME writes a single key–value association (Meng et al., 2022), and MEMIT extends this construction to multiple facts and layers (Meng et al., 2023). Later work improves preservation or identifies parameter subspaces in which updates interfere less with stored knowledge, including PMET and AlphaEdit (Li et al., 2024; Fang et al., 2025). Other editors address continual or aging knowledge and more structured update objectives (Hartvigsen et al., 2023; Ikram et al., 2026). CoRE (Park et al., 2025) studies context sensitivity in a single queried model, showing that preceding context can reactivate pre-edit knowledge. Our work studies a different source of failure: the knowledge request is fixed, but the edited role's input representation shifts because of role instructions and messages from unedited peers.

Multi-agent language-model systems. Frameworks such as AutoGen (Wu et al., 2024), MetaGPT (Hong et al., 2024), and CAMEL (Li et al., 2023) organize language models into role-specialized agents that communicate through structured workflows. Debate, review, and role-playing can improve reasoning by exposing diverse intermediate views (Du et al., 2023; Chan et al., 2023). The same interaction also complicates targeted parameter updates: agents with original parameters may restate outdated knowledge, and the edited role may conform to or be distracted by those messages (Weng et al., 2025; Ju et al., 2025; Ko et al., 2026). Prior work studies conflict resolution and social influence in interacting agents (Chen et al., 2022; Xie et al., 2024). We connect these effects to the internal key on which a parameter edit acts, and study whether editing a single designated agent can provide a useful cost–performance trade-off relative to editing the full system.

6 Conclusion

We studied knowledge editing in MAS, where edits must survive role conditioning, peer communication, and downstream aggregation. We identified interaction-induced representation shift as a key reason why edits built on isolated prompts fail after deployment. To address this problem, we introduced MAKE, which combines adversarial context mining with a closed-form Wasserstein-robust multi-key update. Across models, benchmarks, and communication architectures, MAKE improves system-level editing reliability while preserving generalization, unrelated knowledge, and broader capabilities at practical cost.

7 AI USE STATEMENT

Generative AI tools were used only to aid and polish the writing of this manuscript, including grammar correction, wording refinement and improving readability. All technical content, research hypotheses, experimental designs, results and conclusions were fully conceived, produced and verified by the authors. Every suggestion from the AI tools has been manually reviewed and edited by the authors, who take full responsibility for the entire contents of this submission.

REFERENCES

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. *CoRR*, abs/2308.07201, 2023.

Hung-Ting Chen, Michael J. Q. Zhang, and Eunsol Choi. Rich knowledge sources bring complex knowledge conflicts: Recalibrating models to reflect conflicting evidence. In *EMNLP*, 2022.

Qizhou Chen, Dakan Wang, Taolin Zhang, Zaoming Yan, Chengsong You, Chengyu Wang, and Xiaofeng He. Uniedit: A unified knowledge editing benchmark for large language models. In *NeurIPs*, 2025.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *TACL*, pp. 283–298, 2024.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In *EMNLP*, pp. 6491–6506, 2021.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *CoRR*, abs/2305.14325, 2023.

Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the wasserstein metric: performance guarantees and tractable reformulations. *Math. Program.*, pp. 115–166, 2018.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. AlphaEdit: Null-space constrained knowledge editing for language models. In *ICLR*, 2025.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *EMNLP*, pp. 5484–5495, 2021.

Aaron Grattafiori et al. The llama 3 herd of models, 2024.

Thomas Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with GRACE: Lifelong model editing with discrete key-value adaptors. In *NeurIPs*, 2023.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijian Lin, et al. MetaGPT: Meta programming for a multi-agent collaborative framework. In *ICLR*, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022.

Zarif Ikram, Arad Firouzkouhi, Stephen Tu, Mahdi Soltanolkotabi, and Paria Rashidinejad. CrispEdit: Low-curvature projections for scalable non-destructive LLM editing. *CoRR*, abs/2602.15823, 2026.

Saeid Jamshidi. Collective hallucination in multi-agent llms:modeling and defense. *CoRR*, abs/2606.07941, 2026.

Tianjie Ju, Bowen Wang, Hao Fei, Mong-Li Lee, Wynne Hsu, Yun Li, Qianren Wang, Pengzhou Cheng, Zongru Wu, Zhuosheng Zhang, and Gongshen Liu. Investigating the adaptive robustness with knowledge conflicts in LLM-based multi-agent systems. *CoRR*, abs/2502.15153, 2025.

Changgeon Ko, Jisu Shin, Hoyun Song, Huije Lee, Eui Jun Hwang, and Jong C. Park. Social dynamics as critical vulnerabilities that undermine objective decision-making in LLM collectives. In *ACL*, pp. 37865–37890, 2026.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *CoNLL*, pp. 333–342, 2017.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *NeurIPs*, 2023.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. PMET: Precise model editing in a transformer. In *AAAI*, pp. 18564–18572, 2024.

Zhuo Li, Yupeng Zhang, Pengyu Cheng, Jiajun Song, Mengyu Zhou, Hao Li, Shujie Hu, Yu Qin, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. MARCH: multi-agent reinforced check for hallucination. In *ACL*, pp. 39389–39415, 2026.

Hanbing Liu, Lang Cao, and Yang Li. RAG or learning? understanding the limits of LLM adaptation under continuous knowledge drift in the real world. In *ACL*, 2026.

Shuxin Liu and Jianhao Zhang. CAKE: Causal-guided adaptive knowledge editing for LLMs. In *ACL*, pp. 20040–20070, 2026.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *NeurIPs*, volume 35, pp. 17359–17372, 2022.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *ICLR*, 2023.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. In *ICLR*, 2022.

Haewon Park, Gyubin Choi, Minjun Kim, and Yohan Jo. Context-robust knowledge editing for language models. In *Findings of ACL*, pp. 10360–10385, 2025.

Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Kunlun Zhu, Hanchen Xia, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Scaling large language model-based multi-agent collaboration. In *ICLR*, 2025.

Pingzhi Tang, Yiding Wang, and Muhan Zhang. Knowledge is not enough: Injecting RL skills for continual adaptation. In *ACL*, pp. 11969–11997, 2026.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D. Nguyen. Multi-agent collaboration mechanisms: A survey of llms. *ArXiv*, 2025.

Cedric Villani. *Optimal Transport: Old and New*. Grundlehren der Mathematischen Wissenschaften. Springer, 2009.

Zhiyuan Weng, Guikun Chen, and Wenguan Wang. Do as we do, not as you think: The conformity of large language models. In *ICLR*, 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 conversation. In *ICLR*, 2024.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *ICLR*, 2024.

An Yang et al. Qwen2.5 technical report, 2025.

Grace Hui Yang, Pranav N. Venkit, Hooman Sedghamiz, Enrico Santus, Victor Dibia, and Ioana Baldini. Agents in the wild: Where research meets deployment. *CoRR*, abs/2607.19336, 2026.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In *EMNLP*, pp. 10222–10240, 2023.

A Experimental Setup

A.1 Datasets

We use 2,000 edits from CounterFact (Meng et al., 2022), zsRE (Levy et al., 2017), RippleEdits (Cohen et al., 2024), and UniEdit (Chen et al., 2025).

- **CounterFact:** It evaluates counterfactual editing by replacing the original object o in a factual association (s, r, o) with a low-probability target o^*. It provides paraphrased prompts for generalization, neighborhood prompts for preservation, and generation prompts for consistency and fluency. We use it to test whether edits remain effective and specific under multi-agent communication.
- **zsRE:** It formulates relational knowledge as question answering. Each editing instance contains a source question, a desired answer, a back-translated paraphrase, and an unrelated locality question. It complements CounterFact by testing factual editing and linguistic generalization in a question-answer format.
- **RippleEdits:** It examines whether an edit consistently propagates to its logically implied consequences. Constructed from recent, random, and popular Wikidata facts, it evaluates logical generalization, compositional reasoning, subject aliasing, preservation, and relation specificity. We use it to measure whether edited knowledge remains coherent across related queries and multi-agent interactions.
- **UniEdit:** It is an open-domain benchmark spanning 25 domains in five broad categories. Its neighborhood multi-hop chain sampling procedure derives structurally related generalization and locality samples from each edited triple. By covering rephrasing, multi-hop reasoning, relation reversal, entity aliasing, specificity, and their combinations, UniEdit evaluates editing robustness across diverse domains and complex knowledge dependencies.

CounterFact and zsRE use the edited target for generalization. RippleEdits and UniEdit use their dataset-provided generalization targets. CounterFact preservation is measured against the unedited model's output because its neighborhood prompts have no gold answers. The remaining datasets use their provided preservation answers.

A.2 Evaluation Metrics

To characterize editing performance from isolated acquisition to collaborative deployment, we define five complementary metrics. Let $\mathcal{D}_{\text{eval}}$ be the set of N evaluation edits. For an edit e, let $\mathcal{T}_e^{\text{gen}}$ and $\mathcal{T}_e^{\text{pres}}$ denote its generalization and preservation prompt–reference pairs. Given (q, y) and a response z, $\phi_{q,y}(z) \in \{0, 1\}$ indicates whether z expresses the reference answer y under the benchmark-specific matching rule. For the original edit query, this definition reduces to $\phi_{q_e,o^*} = \phi_e$.

- **Single-Agent Edit Success (SAES).** It evaluates the edited role on the clean query q_e, without role instructions or messages from other agents:

$$\text{SAES} = \frac{1}{N} \sum_{e \in \mathcal{D}_{\text{eval}}} \phi_e(f_{\theta_e + \Delta\theta_e}(q_e)) \,. \tag{19}$$

  It measures target-answer success under isolated inference and provides a clean-query reference for comparison with system-level performance.

- **Multi-Agent Edit Success (MAES).** MAES evaluates whether the final response of the edited multi-agent system expresses the target fact:

$$\text{MAES} = \frac{1}{N} \sum_{e \in \mathcal{D}_{\text{eval}}} \phi_e\Big(\mathcal{M}^{\text{edit}}_{\{a_e\}}(q_e)\Big) \,. \tag{20}$$

  Unlike SAES, MAES includes the complete communication process specified by $\mathcal{G}$ and Π. It averages final-system success over all evaluation edits, without conditioning on single-agent success. $\Delta = \text{SAES} - \text{MAES}$ describes the difference between clean-query and collaborative performance. It is not a conditional survival rate or a causal estimate of deployment-induced loss.

- **Multi-Agent Generalization (MAG).** It measures whether the final system response generalizes the edit to related queries whose answers should change consistently with the edited fact: Let $\mathcal{D}_{\text{gen}} = \{e \in \mathcal{D}_{\text{eval}} : |\mathcal{T}_e^{\text{gen}}| > 0\}$ denote edits with valid generalization tests.

$$\text{MAG} = \frac{1}{|\mathcal{D}_{\text{gen}}|} \sum_{e \in \mathcal{D}_{\text{gen}}} \frac{1}{|\mathcal{T}_e^{\text{gen}}|} \sum_{(q,y) \in \mathcal{T}_e^{\text{gen}}} \phi_{q,y}\Big(\mathcal{M}_{\{a_e\}}^{\text{edit}}(q)\Big). \tag{21}$$

  The inner average assigns equal weight to the generalization queries associated with an edit, while the outer average prevents edits with more test queries from dominating the result. CounterFact and zsRE primarily evaluate paraphrased expressions of the edited association, whereas RippleEdits and UniEdit additionally include dataset-provided logical and compositional consequences.

- **Multi-Agent Preservation (MAP).** MAP evaluates the final system on queries whose reference answers should remain unchanged after editing: Let $\mathcal{D}_{\text{pres}} = \{e \in \mathcal{D}_{\text{eval}} : |\mathcal{T}_e^{\text{pres}}| > 0\}$ denote edits with valid preservation tests.

$$\text{MAP} = \frac{1}{|\mathcal{D}_{\text{pres}}|} \sum_{e \in \mathcal{D}_{\text{pres}}} \frac{1}{|\mathcal{T}_e^{\text{pres}}|} \sum_{(q,y) \in \mathcal{T}_e^{\text{pres}}} \phi_{q,y}\Big(\mathcal{M}_{\{a_e\}}^{\text{edit}}(q)\Big). \tag{22}$$

  For both MAG and MAP, queries are averaged within each edit and then across eligible edits. Edits without valid tests for a metric are excluded from that metric's denominator. The formulas apply to nonempty eligible sets and an entirely empty set provides no evaluable evidence for that metric. MAP captures collateral interference that may be amplified as edited representations propagate through the communication graph. CounterFact neighborhood prompts do not provide gold answers, and thus their references are generated by the unedited model. zsRE, RippleEdits, and UniEdit use their dataset-provided preservation references.

- **Adversarial Conflict Robustness (ACR).** Let $m_{<t_e}^{\text{conf}}$ denote a controlled interaction history in which one or more peer messages explicitly support the outdated answer o. ACR is defined on the response of the edited role under this conflicting context:

$$\text{ACR} = \frac{1}{N} \sum_{e \in \mathcal{D}_{\text{eval}}} \phi_e\big(f_{\theta_e + \Delta\theta_e}\big(p_e, m_{<t_e}^{\text{conf}}, q_e\big)\big). \tag{23}$$

  ACR directly tests whether a_e continues to retrieve o^* when its interaction-conditioned representation is shifted toward evidence supporting o. Because it is measured at the edited role rather than at the final system output, ACR isolates robustness to representation shift before downstream agents can either correct or further amplify the error. All five metrics lie in $[0, 1]$ and are reported as percentages.

All methods are scored with the same deterministic evaluator on the designated answer-bearing response: the clean edited-model response for SAES, the edited role's response for ACR, and the final system response for MAES, MAG, and MAP. Intermediate peer messages are not independently counted as successes.

A.3 Backbone LLMs and Baseline Methods

Our experiments are conducted on two backbone LLMs: Llama-3.1 (8B) (Grattafiori et al., 2024) and Qwen-2.5 (7B) (Yang et al., 2025). We compare our method against several strong model editing baselines via updating parameters.

- **Pre-Edited**. The original model without any parameter updates. We evaluate it under the same prompts and multi-agent communication protocols to establish performance before editing.
- **Fine-Tuning (FT)** (Hu et al., 2022). As a direct gradient-based baseline, FT minimizes the negative log-likelihood of the desired answer by updating model parameters. Despite its simplicity, it can overfit individual requests and interfere with unrelated knowledge.
- **MEND** (Mitchell et al., 2022). Using auxiliary editor networks, MEND transforms fine-tuning gradients into localized low-rank parameter updates. Its meta-learned transformation enables efficient editing while encouraging generalization and locality.

- **ROME** (Meng et al., 2022). Under the associative-memory view of transformer MLPs, ROME inserts a desired key–value association through a rank-one update at a causally identified layer. It provides a canonical clean-key baseline for evaluating interaction-induced representation shift.
- **MEMIT** (Meng et al., 2023). Extending ROME to large edit batches, MEMIT distributes regularized updates across multiple MLP layers. This comparison examines whether multi-layer intervention and increased editing capacity improve robustness under multi-agent interaction.
- **AlphaEdit** (Fang et al., 2025). To reduce interference with previously stored knowledge, AlphaEdit projects parameter updates onto the null space of preserved representations. Its strong sequential-editing performance makes it an important preservation-oriented baseline.
- **CAKE** (Liu & Zhang, 2026). It uses layer-wise causal importance scores to adaptively allocate editing residuals across multiple layers. Its Locate–Weight–Assign–Edit framework replaces uniform residual allocation with causal-guided weighting, aiming to reduce cross-layer interference and preserve unrelated knowledge.
- **CrispEdit** (Ikram et al., 2026). Formulating capability preservation as an explicit constraint, CrispEdit projects parameter updates onto low-curvature directions of the capability-loss landscape. Scalable curvature approximations make it a relevant baseline for assessing capability retention under sequential and multi-agent editing.

A.4 Experimental Implementations

Only the designated role receives an edit and all other agents retain the original model. MAKE mines $M = 20$ stochastic contexts at the configured sampling temperature 0.3, adds one greedy context at temperature 0, and retains $m = 5$ contexts with the greedy interaction fixed as the first anchor. For each baselines with different backbones, we instantiate three communication structures: Pipeline (planner $\rightarrow$ executor $\rightarrow$ reviewer), Debate (debater A $\rightarrow$ debater B $\rightarrow$ judge), and Hierarchical (manager $\rightarrow$ worker 1 / worker 2 $\rightarrow$ manager synthesis). The default edited roles are executor, debater B, and worker 2, respectively [5]. Main-result interactions use one round and greedy decoding.Each experimental run uses a single NVIDIA A800 GPU with 80 GB of memory. Table 4 shows the specific hyperparameters.

A.4.1 Unified factorization and role-specific injection

Each editor returns either a dense update $\Delta W \in \mathbb{R}^{d\times d_m}$ or native low-rank factors (B, A) satisfying $\Delta W = BA$. We use factor rank $r_f = 8$, with $B \in \mathbb{R}^{d\times r_f}$ and $A \in \mathbb{R}^{r_f\times d_m}$, and store the factors in fp16. MEND uses an exact SVD before truncation. Factors are grouped by layer and edit index. For row b of an evaluation batch, let $x_b \in \mathbb{R}^{1\times d_m}$ be the row-form MLP activation and $y_b \in \mathbb{R}^{1\times d}$ its output. If (B_b, A_b) are the factors associated with that row's edit, the hook applies with the edited role is active:

$$y_b \leftarrow y_b + x_b A_b^\top B_b^\top. \tag{24}$$

The role mapping is executor for Pipeline, debater B for Debate, and worker 2 for Hierarchical.

A.4.2 Context construction, caching, and numerical safeguards

The clean key uses the prompt `Question: {prompt}\n\nAnswer:...`. Shifted keys use the complete role instruction, upstream messages, and query in evaluation order. Context-mining cache includes the model, dataset, architecture, edited role, temperatures, context budget, number of generated contexts, query placement, query text, new target, method parameters, and the greedy-anchor hash when applicable.

The update solution is performed in float32. The implemented inverse is $(K^\top C_{\epsilon^*}^{-1} K + \lambda_{\text{ridge}} I_{m+1})^{-1}$ with $\lambda_{\text{ridge}} = 10^{-3}$, increased to 10^{-1} if the system remains singular. Non-finite values are rejected. The theoretical expression in eq. (15) is the $\lambda_{\text{ridge}} \rightarrow 0$ limit. The $m = 0$ update is compared with a direct ROME reference by cosine similarity.

[5] We find through Section B.5 that the editing role cannot be a system evaluation role, nor can it be too far away from it.

Table 4: Shared and MAKE-specific hyperparameters.

Group	Parameter	Value
ROME	fact token	subject last token
ROME	target-value steps / learning rate	25 / 0.5
ROME	target-value loss layer	Llama: 31; Qwen: 27
ROME	weight decay / KL factor	5×10^{-4} / 0.0625
ROME	moment data / samples	Wikipedia / 100,000
Context mining	selected / generated contexts	5 / 20
Context mining	default sampled / greedy temperature	0.3 / 0.0
Context mining	reference layer l_{ref}	Llama: 21; Qwen: 5
Context mining	context-generation batch size	64
Robust update	ϵ^*	eq. (14)
Layer selection	mode / chunk size	key geometry / 4 edits
Deployment	factor rank / query placement	8 / query last
Main evaluation	edits / seeds / rounds	2,000 / 1 / 1
Main evaluation	random seed	42
Held-out evaluation	trajectories per edit	3
Held-out evaluation	architectures / edited-role decoding	Pipeline, Debate, Hierarchical / greedy

A.4.3 TARGET-VALUE OPTIMIZATION

The clean row uses the target value v^* optimized on the clean prompt. Contextual rows use v_g^* optimized on the greedy anchor. $h_{\text{init}} \in \mathbb{R}^d$ denotes the unedited hidden representation at the target-value optimization site and $\Delta v_{\text{opt}} \in \mathbb{R}^d$ the learned additive displacement. We clip this displacement according to

$$\|\Delta v_{\text{opt}}\|_2 \leq \kappa_{\text{arch}} \|h_{\text{init}}\|_2, \tag{25}$$

where κ_{arch} is 2.0, 2.5, and 4.0 for Debate, Hierarchical, and Pipeline. This constraint controls over-activation and repetition collapse.

A.4.4 DATA SELECTION AND PROMPT DISCLOSURE

For each dataset, we evaluate 2,000 usable edit requests. The development procedures, including MEND training, anchor-clamp selection, and ablation tuning, use dataset-specific edit prefixes; these prefixes are not treated as a disjoint held-out split. The background-key moment C is estimated separately from Wikipedia samples. The complete role instructions and message templates are shown in Appendix D. These templates ask agents to state factual details and mark current versus outdated information.

A.4.5 LAYER SELECTION

We set the reference layer l_{ref} to 21 for Llama-3.1-8B and 5 for Qwen2.5-7B (zero-based indices). The admissible MLP-layer sets are $\mathcal{L} = \{17, 18, \ldots, 25\}$ for Llama and $\mathcal{L} = \{1, 2, \ldots, 9\}$ for Qwen. These sets are fixed before context mining. For each edit, eq. (9) selects one layer from the corresponding set, and the update is applied to that layer's MLP `down_proj` weight.

A.4.6 MULTI-AGENT BASELINE ADAPTATIONS

For each edit, $C_{\text{adv}}[0]$ is the complete input to the designated agent on the greedy multi-agent trajectory. Unless stated otherwise, each baseline uses this single interaction context rather than the additional mined contexts used by MAKE. During evaluation, the resulting patch is active only when the designated agent generates; all other agents retain the original model.

ROME. The update direction and key are extracted from a clean factual prompt, while v^* is optimized with $C_{\text{adv}}[0]$ as the context prefix. Thus, the multi-agent interaction conditions the target value but not the clean-prompt update direction or key. This baseline tests context-conditioned target-value optimization without multi-key fitting.

FT. The edit loss is computed from $C_{\mathrm{adv}}[0]$ followed by the desired answer; no clean prompt or additional mined context enters the default training loss. Fine-tuning produces one row-specific rank-8 adapter per edit. We train for 20 steps with learning rate 5×10^{-4} and constrain $\|\Delta W\|$ to at most 3% of $\|W\|$.

MEND. At editing time, MEND computes its input gradient from $C_{\mathrm{adv}}[0]$ and the desired answer, then transforms that gradient into a patch at the selected layer. Editors are trained separately for each dataset–architecture pair. The objective combines edit and locality losses with weights 0.1 and 1.0. We use rank $1{,}920$, 3,000 steps, gradient accumulation 10, outer learning rate 10^{-6}, edit-rate learning rate 10^{-4}, and inference scale 0.25. Signed per-layer edit rates are clipped to $[-10^{-2}, 10^{-2}]$.

MEMIT. MEMIT uses $C_{\mathrm{adv}}[0]$ for target-value optimization, key extraction, and the current representation whose residual is distributed across the edited layers. It preferentially locates the subject in the anchor's query segment. The model-specific default layer sets are 17–24 for Llama and 4–8 for Qwen (zero-based). The single-context Woodbury solve uses covariance weight $15{,}000$. Target-value optimization uses 25 steps, learning rate 0.1, weight decay 0.5, and clamp 0.75.

AlphaEdit. AlphaEdit likewise obtains the target value, layer keys, and current representations from $C_{\mathrm{adv}}[0]$. Its low-variance projectors are computed from clean-model second-moment statistics, rather than from the interaction context. We use threshold 2×10^{-2} and ℓ_2 coefficient 10.0. It edits the same model-specific default layer sets as MEMIT using 25 target-value steps, learning rate 0.1, weight decay 0.4, and clamp 0.5.

CAKE. CAKE uses $C_{\mathrm{adv}}[0]$ together with the AlphaEdit projector and layer-dependent residual weights. For Llama-3.1-8B, the default edited layers are $\{17,\ 18,\ 19,\ 20,\ 21,\ 22,\ 23,\ 24\}$, with the fixed score vector $(0.48,\ 0.43,\ 0.45,\ 0.45,\ 0.49,\ 0.4841,\ 0.50,\ 0.45)$. For Qwen2.5-7B, the default edited layers are $\{4,\ 5,\ 6,\ 7,\ 8\}$, with scores $(0.29,\ 0.29,\ 0.28,\ 0.28,\ 0.27)$, obtained from mean MLP restoration scores on causal-tracing cases. The null-space threshold and ℓ_2 coefficient are 0.02 and 10, respectively, matching our AlphaEdit configuration.

CrispEdit. CrispEdit computes its edit loss solely from $C_{\mathrm{adv}}[0]$ and the desired answer. The K-FAC statistics used to identify low-curvature directions are estimated separately from 1,000 Wikipedia documents; those documents are not additional multi-agent edit contexts. The method updates MLP `down_proj` weights at layers 19–23 for Llama-3.1-8B and 15–19 for Qwen2.5-7B (zero-based). We use a curvature-energy threshold of 0.9, learning rate 5×10^{-4}, at most 25 optimization steps, zero weight decay, and an edit-loss early-stopping threshold of 10^{-2}.

A.4.7 Metric Implementation

Target matching is case-insensitive and uses word boundaries. For each match, the evaluator searches a 20-character prefix and a 45-character suffix for explicit negation. "not only" is excluded from the prefix rule. and empty targets fail. Repetition collapse is detected by any of three rules: (1) at least three identical consecutive words; (2) a bigram occurring at least three times whose neighboring words are each at least 70% dominated by a member of that bigram; (3) an output of at least eight words in which one word accounts for at least 50% of tokens and appears in an adjacent repetition. A collapsed output fails even if it contains the target. The matcher rejects target occurrences in detected negated spans and repetition-collapsed responses, although quotation- and paraphrase-related ambiguity remains a limitation of this lexical evaluation protocol. CounterFact neighborhood prompts have no gold answers. The unedited model therefore generates their references before editing, and a cleaner removes trailing punctuation and common answer-introduction text. zsRE uses its locality answer field. RippleEdits draws MAG tests from Logical Generalization, Compositionality I/II, and Subject Aliasing, and MAP tests from Relation Specificity and Forgetfulness. UniEdit reads the targets attached to its generality and locality records.

Table 5: Main results with Qwen2.5 (7B). S/M/A/G/P denote SAES/MAES/ACR/MAG/MAP. Bold and underlined numbers mark the best and second-best results.

Method	CounterFact					zsRE					RippleEdits					UniEdit					Avg.
	S	M	A	G	P	S	M	A	G	P	S	M	A	G	P	S	M	A	G	P	
Pre-Edited	1.4	2.2	2.1	1.3	26.8	21.4	19.6	21.1	17.6	20.2	0.3	0.3	0.4	3.6	26.1	6.8	9.1	9.5	4.0	6.6	10.0
FT	32.2	32.4	41.9	16.0	25.6	42.5	42.3	45.4	36.0	19.2	5.1	7.7	6.4	6.7	**24.4**	15.9	27.8	20.9	4.2	5.8	22.9
MEND	43.4	19.3	39.1	12.1	26.2	29.3	25.3	27.4	22.7	19.9	4.5	1.8	3.2	4.9	24.3	14.2	12.5	12.0	3.4	5.9	17.6
ROME	91.9	55.6	54.1	37.1	26.4	74.1	54.7	64.7	48.5	20.1	54.6	33.7	39.5	18.0	22.0	47.6	37.4	32.4	5.5	6.1	41.2
MEMIT	55.1	47.7	61.2	18.8	**26.6**	46.1	49.7	47.7	35.1	20.1	15.2	24.0	19.9	9.1	23.7	22.1	30.6	21.7	3.9	5.9	29.2
AlphaEdit	64.1	55.0	71.9	21.5	26.5	56.8	59.8	59.8	43.2	**20.3**	28.7	36.9	39.3	13.2	23.2	30.2	28.6	29.7	4.5	6.1	36.0
CAKE	66.2	54.2	74.7	22.3	25.8	57.5	59.6	60.4	43.7	18.4	29.5	36.5	39.1	13.3	23.3	30.3	33.9	29.9	4.5	6.1	36.5
CrispEdit	29.8	52.2	75.0	39.8	15.6	49.1	60.1	64.7	53.5	15.7	14.8	34.6	**40.2**	23.3	18.0	41.0	37.8	**35.8**	6.4	6.4	35.7
MAKE	**94.0**	**56.5**	**84.1**	**40.1**	26.1	**82.5**	**60.2**	**66.3**	**53.7**	**20.3**	**76.8**	**37.3**	**40.2**	**24.1**	23.2	**63.1**	**38.0**	32.7	**6.6**	**6.7**	**46.6**

Table 6: Results on four benchmarks under different multi-agent communication architectures with Llama 3.1 (8B). S/M/A/G/P denote SAES/MAES/ACR/MAG/MAP.

Architecture	Method	CounterFact					zsRE					RippleEdits					UniEdit				
		S	M	A	G	P	S	M	A	G	P	S	M	A	G	P	S	M	A	G	P
Llama 3.1 (8B)																					
Pipeline	Pre-Edited	1.1	2.1	2.6	1.7	27.6	26.3	27.5	33.5	26.4	21.1	0.2	0.1	0.4	3.6	22.8	9.0	10.5	13.2	3.8	5.5
	FT	37.0	42.3	89.9	32.5	14.2	79.0	76.3	85.0	72.5	18.1	14.6	27.1	35.0	19.2	11.1	62.4	55.6	67.9	10.4	6.2
	MEND	35.5	36.4	65.5	27.7	22.4	50.0	49.9	55.5	48.8	20.3	19.1	24.5	32.1	16.0	18.9	38.5	36.4	40.8	7.7	5.8
	ROME	84.9	18.3	54.4	14.7	26.0	86.8	56.7	74.4	48.0	20.8	40.0	5.7	25.2	4.9	21.4	71.0	31.5	45.1	5.0	5.3
	MEMIT	3.2	46.0	28.6	4.3	27.5	37.5	70.9	53.1	40.9	20.9	0.7	10.4	2.5	4.6	23.1	16.6	40.9	25.0	4.2	5.3
	AlphaEdit	3.4	51.2	32.6	4.9	27.4	37.0	76.5	52.3	40.8	20.8	0.6	31.2	3.5	6.9	23.0	17.0	54.2	26.9	4.2	5.2
	CAKE	3.4	51.1	34.1	5.1	27.6	37.0	76.8	52.4	40.9	20.8	0.7	31.9	3.5	7.3	22.8	17.2	54.9	27.1	4.1	5.1
	CrispEdit	88.2	53.2	81.3	36.1	14.9	77.6	71.2	67.7	69.0	15.2	48.9	39.7	60.1	31.5	11.2	63.3	47.0	47.1	11.0	6.3
	MAKE	97.0	58.0	91.5	49.1	22.7	96.5	81.7	87.6	70.7	20.8	91.1	46.6	61.4	30.1	17.9	93.3	65.8	72.7	9.4	6.3
Debate	Pre-Edited	1.1	1.6	5.2	1.2	33.9	26.3	21.6	32.4	18.2	21.9	0.2	0.1	0.4	2.9	17.9	9.0	9.2	12.4	2.5	3.7
	FT	13.1	26.2	70.1	19.0	33.0	51.9	54.9	64.4	49.3	20.2	1.7	15.2	16.1	13.6	23.4	38.0	39.4	52.2	7.1	5.8
	MEND	32.2	24.7	71.2	26.1	32.0	48.4	43.0	53.6	38.9	21.2	10.8	15.8	22.1	12.5	25.2	35.9	31.9	42.2	6.8	6.2
	ROME	79.3	7.1	51.1	4.8	29.3	75.2	38.7	62.4	26.7	19.9	38.8	6.7	32.6	3.8	22.6	63.0	23.7	41.7	4.0	4.8
	MEMIT	2.8	23.9	28.7	2.0	29.6	35.5	55.8	47.6	23.2	19.9	0.9	14.9	6.6	3.9	21.2	14.8	28.2	21.7	3.2	5.0
	AlphaEdit	2.9	22.7	29.8	2.3	29.6	35.7	57.0	48.4	25.1	19.2	0.9	23.2	7.8	5.9	21.3	15.6	36.3	22.5	3.4	5.2
	CAKE	2.9	22.3	30.1	2.4	29.9	35.9	57.8	48.6	24.9	20.3	1.1	22.4	8.0	5.5	21.8	15.8	36.7	22.7	3.2	5.2
	CrispEdit	68.4	21.4	64.5	34.9	35.5	57.7	56.6	54.4	54.8	20.7	12.7	24.6	59.9	23.0	30.0	48.0	38.3	47.0	9.0	5.8
	MAKE	96.9	25.1	84.5	26.0	34.3	94.6	57.7	78.7	47.7	20.6	90.6	26.6	61.5	20.0	27.8	92.5	42.8	63.9	7.1	6.3
Hierarchical	Pre-Edited	1.1	0.9	2.8	0.6	30.9	26.3	21.4	29.0	20.7	22.9	0.2	0.2	0.4	3.7	26.5	9.0	8.6	10.9	2.3	5.1
	FT	9.0	25.4	52.9	13.2	25.9	44.7	48.2	60.5	42.8	19.6	1.5	9.7	16.8	10.7	25.2	35.6	38.6	52.8	6.7	6.3
	MEND	25.4	23.5	61.8	17.9	30.5	50.3	42.4	54.6	37.8	21.0	12.2	11.1	23.8	11.0	27.9	24.3	19.1	25.9	4.8	5.1
	ROME	75.2	6.2	40.4	4.2	34.4	75.7	34.6	59.3	26.6	22.2	34.1	5.1	25.1	5.0	28.0	64.5	21.9	40.4	4.7	5.3
	MEMIT	2.2	22.7	15.0	1.6	33.9	33.0	44.8	40.2	22.9	22.8	0.4	6.3	1.5	4.8	28.1	14.8	27.9	18.8	3.6	5.1
	AlphaEdit	2.4	26.5	20.9	1.9	33.9	33.3	51.8	40.9	23.8	22.4	0.4	16.0	2.6	6.5	27.7	15.8	39.2	20.8	3.9	5.5
	CAKE	2.4	26.2	21.9	2.1	33.6	33.4	52.6	40.9	23.9	22.6	0.4	15.4	2.8	7.1	27.7	16.0	37.0	20.9	3.5	5.1
	CrispEdit	56.5	25.7	79.2	28.0	24.6	48.9	55.0	67.3	51.0	19.2	9.4	20.6	62.1	11.7	25.2	43.0	38.2	48.6	7.1	6.3
	MAKE	96.6	26.5	79.3	25.2	31.2	94.9	57.0	76.5	46.7	22.4	90.9	23.4	61.6	18.9	26.5	92.7	47.6	64.7	6.9	5.9

B Other Experiments

B.1 Editing and General Ability Results

Table 5 reports the overall results on Qwen2.5-7B across CounterFact, zsRE, RippleEdits, and UniEdit. The five metrics assess single-agent and system-level editing success, resistance to conflicting information, generalization to related queries, and preservation of unrelated knowledge, complementing the Llama-3.1-8B results in the main text.

Figure 6 reports the general-capability results on RTE, MMLU, and NLI, complementing the SST, MRPC, and CoLA results in Figure 3. We measure macro-F1 after 0, 10, 50, 100, and 200 cumulative edits. These curves characterize how general capabilities change as edits accumulate, with the zero-edit checkpoint providing the initial performance reference.

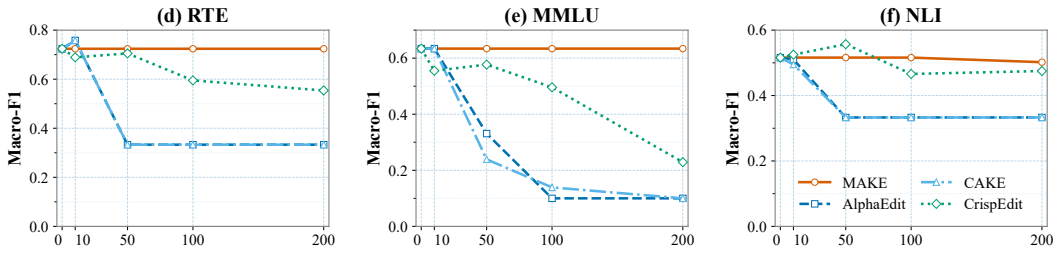

Figure 6: General-capability performance on RTE, MMLU, and NLI after 0, 10, 50, 100, and 200 cumulative edits. The horizontal axis indicates the number of edits; the vertical axis reports macro-F1.

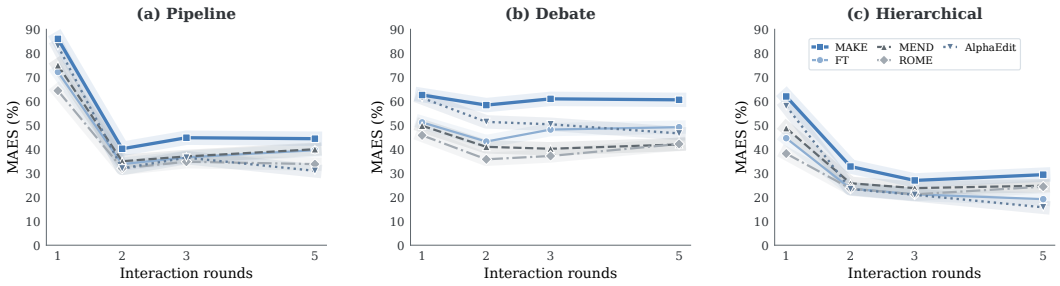

Figure 7: Multi-round editing performance on zsRE across Pipeline, Debate, and Hierarchical architectures. Each interaction executes exactly 1, 2, 3, or 5 rounds. Shaded strokes are visual guides, not confidence intervals.

B.2 Different Architecture Results

Tables 6 and 7 report architecture-specific results for Llama-3.1-8B and Qwen2.5-7B, respectively, under Pipeline, Debate, and Hierarchical communication.

B.3 Edit Survival under Multi-Round Interaction

Figures 4 and 7 report multi-round editing performance on 500 CounterFact and 500 zsRE edits, respectively. We compare MAKE, FT, MEND, ROME, and AlphaEdit under Pipeline, Debate, and Hierarchical communication. Each query executes exactly 1, 2, 3, or 5 interaction rounds, and MAES measures editing success in the final team response.

B.4 Computational Cost and Context Scalability

Table 8 extends the computational cost analysis by varying the number of retained interaction contexts over $m \in \{0, 5, 10, 20\}$. The average score is the arithmetic mean of SAES, MAES, ACR, MAG, and MAP.

B.5 Role-Location Sensitivity

We study how the location of a single MAKE intervention affects editing performance and preservation. For each architecture, we fix $|\mathcal{S}| = 1$ and edit each role in turn, keeping all other settings unchanged. We report MAES on the final team response and MAP on the same preservation queries, together with their sum (Total). Table 10 shows a clear MAES–MAP trade-off. Editing an early upstream role gives relatively high preservation but low MAES, because the corrected fact must pass through more unedited agents and can be weakened or overwritten. Moving the edit closer to the final output usually strengthens its effect on the team answer, but editing the final aggregator also lowers MAP because any collateral change directly affects the system response. The roles used in our main experiments occupy an effective middle position: the executor precedes the reviewer, debater B precedes the judge, and workers 1 and 2 are close to manager synthesis, with worker 2

Table 7: Results on four benchmarks under different multi-agent communication architectures with Qwen 2.5 (7B). S/M/A/G/P denote SAES/MAES/ACR/MAG/MAP.

Architecture	Method	CounterFact					zsRE					RippleEdits					UniEdit				
		S	M	A	G	P	S	M	A	G	P	S	M	A	G	P	S	M	A	G	P
Qwen 2.5 (7B)																					
Pipeline	Pre-Edited	1.4	1.5	2.2	1.7	28.5	21.4	19.9	24.0	18.5	19.1	0.3	0.2	0.4	3.6	27.3	6.8	9.3	12.4	3.5	5.6
	FT	61.4	61.5	74.8	35.3	26.6	58.0	64.0	68.0	56.3	17.5	12.7	19.5	15.7	11.2	23.0	24.2	46.5	35.0	5.2	5.9
	MEND	45.8	32.3	44.1	22.0	27.4	35.3	30.0	34.7	28.9	19.1	6.3	3.5	5.2	6.2	21.9	17.3	17.7	16.6	3.4	5.7
	ROME	95.3	71.6	58.0	48.4	27.9	81.2	71.1	75.8	65.8	18.6	72.1	57.0	43.4	28.6	19.2	55.3	56.0	42.1	7.8	5.9
	MEMIT	73.7	72.6	81.7	38.7	28.6	56.4	72.2	64.8	54.9	18.7	28.9	50.1	38.0	16.3	22.0	31.6	51.6	30.3	4.7	5.8
	AlphaEdit	73.9	70.8	82.3	39.5	28.3	66.7	68.7	74.8	62.7	18.9	43.4	44.4	54.5	22.1	21.2	39.6	48.5	37.0	5.9	6.0
	CAKE	77.4	73.7	86.0	40.9	28.2	67.6	67.7	76.0	63.5	19.1	44.7	44.4	55.1	21.6	21.4	39.2	53.5	35.8	5.9	5.9
	CrispEdit	55.5	65.8	74.2	45.5	12.0	71.4	64.8	76.4	57.6	12.2	28.8	44.3	48.7	29.2	11.1	62.8	44.5	42.0	6.3	6.3
	MAKE	94.5	71.6	87.8	49.8	27.8	84.8	72.4	74.6	64.8	18.8	78.0	57.1	56.5	30.7	20.6	66.7	52.7	39.0	8.3	7.0
Debate	Pre-Edited	1.4	2.4	2.3	1.1	22.9	21.4	17.3	23.3	15.6	19.0	0.3	0.4	0.3	3.4	22.3	6.8	8.5	10.0	5.1	8.0
	FT	18.9	17.8	27.0	6.4	24.0	33.1	31.7	33.4	24.2	19.6	1.6	2.2	2.5	4.3	24.8	11.5	18.9	15.7	3.3	6.1
	MEND	39.6	13.5	35.8	8.8	23.8	29.4	24.3	28.4	20.3	19.2	3.9	1.2	2.6	4.5	22.6	12.8	10.2	12.9	3.4	5.7
	ROME	90.8	50.8	66.3	34.4	23.0	72.8	49.1	56.6	41.2	19.6	56.4	29.0	40.2	15.6	20.1	42.7	29.2	35.8	4.2	5.4
	MEMIT	44.2	38.6	49.3	10.4	22.9	39.6	41.0	41.8	25.2	19.6	11.8	15.6	14.2	6.2	19.9	16.8	21.4	19.3	3.3	5.4
	AlphaEdit	57.0	49.7	63.5	14.3	22.9	50.7	54.0	52.9	32.3	19.8	26.4	38.0	33.7	8.7	20.5	23.8	23.3	27.2	3.6	5.6
	CAKE	59.2	44.4	67.5	14.6	21.2	51.6	53.7	52.9	32.0	20.0	26.8	37.5	33.4	9.5	20.3	24.4	28.8	27.8	3.5	5.8
	CrispEdit	14.6	52.9	79.5	42.0	13.8	38.8	57.7	67.3	50.3	14.8	7.5	34.4	39.0	21.7	17.6	31.0	33.7	32.7	6.3	6.3
	MAKE	93.5	52.9	82.9	38.5	22.5	80.3	59.6	69.2	54.2	19.8	76.6	31.0	34.1	21.2	20.8	59.6	29.3	29.5	5.8	6.5
Hierarchical	Pre-Edited	1.4	2.8	1.7	1.2	29.0	21.4	21.6	16.1	18.8	22.4	0.3	0.3	0.4	3.9	28.7	6.8	9.4	6.0	3.4	6.2
	FT	16.4	17.8	23.8	6.3	26.1	36.5	31.1	34.8	27.5	20.6	0.9	1.3	1.1	4.5	25.4	12.0	18.1	12.0	4.2	5.3
	MEND	44.7	12.2	37.4	5.5	27.3	23.1	21.5	19.1	18.9	21.4	3.4	0.7	1.9	4.1	28.3	12.4	9.5	6.6	3.5	6.2
	ROME	89.5	44.5	38.0	28.5	28.3	68.2	43.8	61.6	38.6	22.2	35.4	15.1	34.8	9.8	26.7	44.8	27.0	19.3	4.6	6.9
	MEMIT	47.3	31.9	52.5	7.4	28.2	42.4	35.9	36.6	25.1	21.9	4.9	6.3	7.4	4.8	29.3	17.8	18.8	15.5	3.7	6.5
	AlphaEdit	61.5	44.6	70.0	10.7	28.2	53.1	56.6	51.8	34.5	22.1	16.3	28.4	29.8	8.7	27.8	27.1	14.1	25.0	3.9	6.7
	CAKE	62.0	44.6	70.5	11.4	28.0	53.4	57.5	52.3	35.5	16.2	16.9	27.5	28.8	8.8	28.3	27.2	19.5	26.1	4.2	6.5
	CrispEdit	19.3	38.0	71.4	32.0	21.1	37.1	57.7	50.5	52.5	20.1	8.2	25.2	33.0	19.1	25.2	29.1	35.1	32.7	6.7	6.7
	MAKE	94.0	44.9	81.5	32.0	27.9	82.3	48.6	55.1	42.2	22.4	75.9	23.9	30.0	20.3	28.3	62.9	32.1	29.5	5.6	6.5

Table 8: Editing cost of scalability. Avg. score is the mean of all performance metrics. ♣ and † denote uncached and cached runs. Cached runs reuse both context and patch caches.

Method / setting	Mining (s)	Update (s)	Total edit (s)	Infer (s)	Infer Overhead	Peak infer. memory (GB)	Avg. score (%)
♣ MAKE, $m = 0$	0.02	0.379	0.396	3.683	4.65%	19.29	38.8
† MAKE, $m = 0$	0.02	0.001	0.017	3.870	0.85%	18.30	38.8
♣ MAKE, $m = 5$	17.18	1.448	18.625	3.599	2.07%	19.31	63.1
† MAKE, $m = 5$	0.14	0.002	0.143	3.527	2.05%	18.32	63.1
♣ MAKE, $m = 10$	14.71	1.447	16.154	3.487	1.91%	19.32	64.5
† MAKE, $m = 10$	0.25	0.002	0.255	3.447	2.06%	18.33	64.5
♣ MAKE, $m = 20$	15.80	1.464	17.266	3.473	1.97%	19.37	63.6
† MAKE, $m = 20$	0.49	0.005	0.493	3.465	2.15%	18.52	63.6

giving the best overall balance. This analysis supports our designated role choices under a fixed single-agent budget; it does not address per-query role selection.

B.6 HELD-OUT PEER-MESSAGE EVALUATION

The greedy trajectory is included during editing as a deterministic deployment anchor, so we test whether MAKE remains effective when peer messages differ from all edit-time contexts. We evaluate this held-out generalization on CounterFact with Llama 3.1-8B across the three MAS architectures. Each edit is constructed from the clean query, the greedy trajectory, and four selected high-shift contexts; the resulting patch and all hyperparameters are then frozen. We evaluate the same patch on the original greedy trajectory and on three independently sampled peer-message trajectories. Unseen-MAES is averaged over the three samples for each edit, with $\Delta = \text{Greedy-MAES} - \text{Unseen-MAES}$ and Coverage denoting the proportion of valid held-out trajectories.

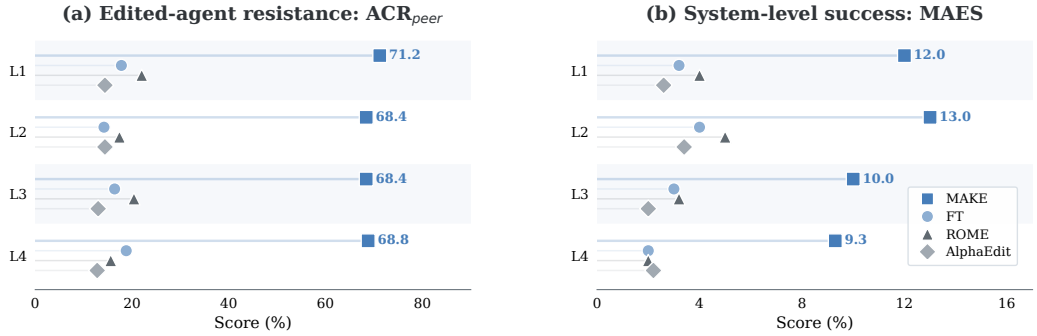

Figure 8: Robustness as peer support for the outdated fact in different prompt–level settings.

Table 9 shows almost no overall degradation under held-out peer messages. Average MAES changes from 33.33 on the greedy trajectory to 33.31 on unseen trajectories, with an average Δ of only 0.02 points and 100% coverage. Pipeline and Hierarchical show only small decreases, while Debate performs slightly better on the sampled trajectories. Since none of these messages appears in the editing contexts, the near-identical performance is unlikely to come from direct context reuse. The frozen update instead generalizes to previously unseen peer messages within the same model, architecture, and prompting setup.

Table 9: Evaluation of the same frozen MAKE edits on the nominal greedy trajectory and on three independently generated held-out peer-message trajectories per edit.

Architecture	Greedy (MAES)	Unseen (MAES)	$\Delta\downarrow$	Coverage
Pipeline	56.20	55.00	1.20	100.00
Debate	21.60	24.20	-2.60	100.00
Hierarchical	22.20	20.73	1.47	100.00
Avg.	33.33	33.31	0.02	100.00

B.7 Scaling Adversarial Peer Strength

The standard ACR evaluation uses a single fixed conflict, so it does not show how an edited role behaves as peer pressure becomes more coordinated and explicit. We construct four controlled conditions on 500 CounterFact edits under the Hierarchical architecture: (1) L1 replaces Worker 1's message with naturally generated support for the pre-edit answer o; (2) L2 makes Worker 1 explicitly insist that o should be retained; (3) L3 has both the Manager and Worker 1 provide consistent support for o; and (4) L4 has both peers provide supporting arguments while rejecting alternative answers. The L4 arguments are model-generated stress-test messages, not externally verified evidence. The prompts for L1–L4 are listed in Appendix D.1. The edited Worker 2 and its stored update remain fixed across all conditions; only the peer messages change.

Table 10: Role-location sensitivity of MAKE under the fixed single-agent budget $|\mathcal{S}| = 1$.

Architecture	Edited role	MAES	MAP	Total
Pipeline	planner	9.2	27.3	36.5
	executor	53.8	17.8	71.6
	reviewer	55.2	14.6	69.8
Debate	debater A	20.6	29.9	50.5
	debater B	26.0	29.6	55.6
	judge	29.8	24.7	54.5
Hierarchical	manager	19.8	31.7	51.5
	worker 1	27.7	31.0	58.7
	worker 2	29.6	30.7	60.3
	manager syn.	32.3	26.1	58.4

Figure 8 shows that MAKE achieves the highest ACR and MAES at every pressure level. Its ACR remains nearly stable from L1 to L4, which indicates that the edited Worker 2 continues to retrieve o^* even when peers repeatedly reinforce o. MAES is more sensitive and declines under the stronger L3 and L4 conditions, although MAKE retains a clear advantage over all baselines. The gap between stable ACR and decreasing MAES arises because the final manager remains unedited, so repeated support for o can dilute or override Worker 2's corrected response during synthesis. These results show that adversarial context mining and robust multi-key editing improve resistance to interaction-induced shifts, while retrieval by the edited agent and survival through downstream aggregation remain distinct challenges.

C Derivations and Proofs

This appendix gives the derivations behind Representation Shift Theory and the two phases of MAKE. Appendix C.1 and Appendix C.2 prove the contextual attenuation and uncovered-direction results used in Adversarial Context Mining. Appendix C.4 proves the low-dimensional population-shift result and clarifies how the mined covariance is used as an empirical diagnostic. Appendix C.5 and Appendix C.6 prove the Wasserstein energy upper bound, the closed-form constrained update, and the conditional sensitivity bound for a specified off-support deviation. All analyses are carried out at the edited MLP layer; the effect of the retrieved value on token probabilities is evaluated empirically.

C.1 Derivation of the contextual edit decomposition

Proof of theorem 2.1. Define the clean value residual as $r_v = v^* - Wk$. The rank-one update in eq. (3) can be written as

$$\Delta W = r_v a_k^\top, \quad a_k = \frac{C^{-1}k}{k^\top C^{-1}k}. \tag{26}$$

It satisfies the clean constraint because

$$\begin{aligned} \Delta W k &= r_v a_k^\top k = r_v \frac{k^\top C^{-1}k}{k^\top C^{-1}k} = r_v, \\ \widehat{W}k &= Wk + \Delta W k = Wk + r_v = v^*. \end{aligned} \tag{27}$$

For the interaction-conditioned key $k_c = k + \delta(c)$,

$$\begin{aligned} \Delta W k_c &= r_v a_k^\top k_c \\ &= r_v \frac{k^\top C^{-1}k_c}{k^\top C^{-1}k} \\ &= (v^* - Wk)\eta(c), \end{aligned} \tag{28}$$

which proves the first identity in eq. (4). Substituting the shifted key into the edited matrix gives

$$\begin{aligned} \widehat{W}k_c - v^* &= (W + \Delta W)(k + \delta(c)) - v^* \\ &= \underbrace{(W + \Delta W)k - v^*}_{=0} + W\delta(c) + \Delta W \delta(c) \\ &= W\delta(c) + \Delta W(k_c - k) \\ &= W\delta(c) + r_v\eta(c) - r_v \\ &= W\delta(c) + (v^* - Wk)(\eta(c) - 1), \end{aligned} \tag{29}$$

which proves the second identity in eq. (4).

Let $a = W\delta(c)$ and $b_c = (v^* - Wk)(\eta(c) - 1)$. The triangle inequality gives

$$\begin{aligned} \|\widehat{W}k_c - v^*\|_2 &= \|a + b_c\|_2 \\ &\le \|a\|_2 + \|b_c\|_2 \\ &= \|W\delta(c)\|_2 + \|v^* - Wk\|_2 |\eta(c) - 1|, \end{aligned} \tag{30}$$

and the reverse triangle inequality gives

$$\begin{aligned} \|\widehat{W}k_c - v^*\|_2 &= \|a + b_c\|_2 \\ &\ge \big|\|a\|_2 - \|b_c\|_2\big| \\ &= \big|\|W\delta(c)\|_2 - \|v^* - Wk\|_2 |\eta(c) - 1|\big|. \end{aligned} \tag{31}$$

These inequalities give the two-sided residual bound summarized in the main text. □

Whitened-key interpretation. Because $C \succ 0$, define $\widetilde{k} = C^{-1/2}k$, $\widetilde{k}_c = C^{-1/2}k_c$, and $\widetilde{\delta}(c) = C^{-1/2}\delta(c)$. Then

$$\eta(c) = \frac{k^\top C^{-1} k_c}{k^\top C^{-1} k} = \frac{\widetilde{k}^\top \widetilde{k}_c}{\widetilde{k}^\top \widetilde{k}} = \frac{\|\widetilde{k}_c\|_2}{\|\widetilde{k}\|_2} \cos(\widetilde{k}, \widetilde{k}_c), \tag{32}$$

which proves eq. (32). Equivalently,

$$\eta(c) - 1 = \frac{\widetilde{k}^\top \widetilde{\delta}(c)}{\|\widetilde{k}\|_2^2}. \tag{33}$$

Only the component of the whitened shift parallel to the clean key changes the scalar edit coefficient. An orthogonal shift can leave $\eta(c) = 1$ while still producing the backbone term $W\delta(c)$.

C.2 Derivation for uncovered low-rank directions

Proof of theorem 2.2. Using the compact singular value decomposition in Section 2.3,

$$\Delta W = U_r S_r V_r^\top, \quad U_r^\top U_r = I_r, \quad V_r^\top V_r = I_r. \tag{34}$$

The projection matrices onto the responsive subspace and its orthogonal complement are

$$P_r = V_r V_r^\top, \quad P_r^\perp = I_{d_m} - V_r V_r^\top. \tag{35}$$

For a fixed interaction context c, the shift $\delta(c)$ decomposes as

$$\delta_\parallel(c) = P_r \delta(c), \quad \delta_\perp(c) = P_r^\perp \delta(c), \quad \delta(c) = \delta_\parallel(c) + \delta_\perp(c). \tag{36}$$

Because $V_r^\top V_r = I_r$,

$$\begin{aligned} \Delta W \delta_\perp(c) &= U_r S_r V_r^\top (I_{d_m} - V_r V_r^\top)\delta(c) \\ &= U_r S_r (V_r^\top - V_r^\top V_r V_r^\top)\delta(c) \\ &= U_r S_r (V_r^\top - I_r V_r^\top)\delta(c) = 0. \end{aligned} \tag{37}$$

Thus $\delta_\perp(c) \in \ker(\Delta W)$. Using $k_c = k + \delta(c)$, the clean constraint $\widehat{W}k = v^*$, and eq. (36),

$$\begin{aligned} \widehat{W}k_c - v^* &= \widehat{W}k - v^* + \widehat{W}\delta_\parallel(c) + \widehat{W}\delta_\perp(c) \\ &= \widehat{W}\delta_\parallel(c) + W\delta_\perp(c) + \Delta W \delta_\perp(c) \\ &= \widehat{W}\delta_\parallel(c) + W\delta_\perp(c), \end{aligned} \tag{38}$$

which proves eq. (5). The reverse triangle inequality gives

$$\|\widehat{W}k_c - v^*\|_2 \geq \|W\delta_\perp(c)\|_2 - \|\widehat{W}\delta_\parallel(c)\|_2. \tag{39}$$

If $\delta(c) = \delta_\perp(c)$, then

$$\|\widehat{W}k_c - v^*\|_2 = \|W\delta(c)\|_2. \tag{40}$$

If $r < d_m$, define

$$\mu_\perp = \inf_{\substack{u \in \ker(\Delta W) \\ \|u\|_2 = 1}} \|Wu\|_2.$$

For a nonzero shift satisfying $\delta(c) = \delta_\perp(c)$, let $u = \delta(c)/\|\delta(c)\|_2$; then $u \in \ker(\Delta W)$ and $\|u\|_2 = 1$. Therefore,

$$\|W\delta(c)\|_2 = \|\delta(c)\|_2 \|Wu\|_2 \geq \mu_\perp \|\delta(c)\|_2. \tag{41}$$

The same inequality is immediate when $\delta(c) = 0$. If $r = d_m$, then $\ker(\Delta W) = \{0\}$ and no nonzero pure-null shift exists. □

Rank–nullity also gives

$$\dim \ker(\Delta W) = d_m - \operatorname{rank}(\Delta W) = d_m - r. \tag{42}$$

This shows why the distribution of interaction shifts, not just the ambient dimension, determines whether a compact multi-key edit can succeed.

C.3 Derivation of multi-hop propagation

Proof of proposition 2.3. For $t = 1, \ldots, L$, define the all-stage survival event $H_t = \bigcap_{j=1}^{t} G_j$. By the chain rule and the definition of F_t,

$$\Pr(H_L) = \Pr(G_1) \prod_{t=2}^{L} \Pr(G_t \mid H_{t-1}) = \Pr(E_{\text{agent}}(m_{<t_e})) \prod_{t=2}^{L} F_t. \tag{43}$$

If failure is absorbing, then

$$G_L \subseteq G_{L-1} \subseteq \cdots \subseteq G_1, \tag{44}$$

so $H_L = G_L = E_{\text{system}}$. Hence,

$$\Pr(E_{\text{system}}) = \Pr(E_{\text{agent}}(m_{<t_e})) \prod_{t=2}^{L} F_t, \tag{45}$$

which proves $\Pr(E_{\text{system}})$. If repair is possible but correctness at every stage is sufficient for final system success, then $H_L \subseteq E_{\text{system}}$, and therefore

$$\Pr(E_{\text{system}}) \geq \Pr(H_L) = \Pr(E_{\text{agent}}(m_{<t_e})) \prod_{t=2}^{L} F_t, \tag{46}$$

which proves $\Pr(E_{\text{system}})$ lower bound. □

For a directed acyclic communication graph, let $\mathcal{R}$ be the set of complete directed paths from a_e to the agent producing the system answer. For each $\rho \in \mathcal{R}$, let H_ρ be the event that the edited fact is correct at every stage on path ρ. If $H_\rho \subseteq E_{\text{system}}$ for every $\rho \in \mathcal{R}$, then no independence across paths is required for the bound

$$\Pr(E_{\text{system}}) \geq \max_{\rho \in \mathcal{R}} \Pr(H_\rho). \tag{47}$$

C.4 Derivation of low-dimensional interaction shift

Define the hidden-to-key map $g_l(h) = \sigma(W_l^{\text{in}} h)$, so that $k_l(x) = g_l(h_l(x))$. The population shift statistics under $c \sim \mathcal{P}_c$ are

$$\mu_\delta = \mathbb{E}_{c \sim \mathcal{P}_c}[\delta(c)], \qquad \Sigma_\delta = \mathbb{E}_{c \sim \mathcal{P}_c}\left[(\delta(c) - \mu_\delta)(\delta(c) - \mu_\delta)^\top\right]. \tag{48}$$

The quantities $\widehat{\mu}_\delta$ and $\widehat{\Sigma}_\delta$ in eq. (10) summarize the shifts selected by adversarial context mining. Because selection depends on shift magnitude, they characterize the mined support instead of providing unbiased estimates of eq. (48). The method uses $\widehat{\Sigma}_\delta$ and its plug-in effective dimension $\widehat{d}_{\text{eff}}$ as a stress-oriented empirical geometry in eq. (14), with the zero-covariance convention from eq. (6).

The contextual-factor assumption in theorem 2.4 is

$$h_l(x_c) - h_l(q_e) = B_{\text{ctx}} \zeta(c) + b_{\text{ctx}}, \qquad B_{\text{ctx}} \in \mathbb{R}^{d \times r_c}, \quad \zeta(c) \in \mathbb{R}^{r_c}, \quad b_{\text{ctx}} \in \mathbb{R}^d. \tag{49}$$

Proof of theorem 2.4. Let $J_l \in \mathbb{R}^{d_m \times d}$ be the linear part of g_l on the local region in eq. (49), and define $\bar{\zeta} = \mathbb{E}_{c \sim \mathcal{P}_c}[\zeta(c)]$. Under the affine key-map assumption,

$$\delta(c) = J_l(B_{\text{ctx}} \zeta(c) + b_{\text{ctx}}). \tag{50}$$

Its mean is

$$\mu_\delta = J_l B_{\text{ctx}} \bar{\zeta} + J_l b_{\text{ctx}}, \tag{51}$$

and its centered form is

$$\delta(c) - \mu_\delta = J_l B_{\text{ctx}} (\zeta(c) - \bar{\zeta}). \tag{52}$$

Define the latent covariance

$$\Sigma_\zeta = \mathbb{E}_{c \sim \mathcal{P}_c}\left[(\zeta(c) - \bar{\zeta})(\zeta(c) - \bar{\zeta})^\top\right] \in \mathbb{R}^{r_c \times r_c}. \tag{53}$$

Then

$$\begin{aligned}\Sigma_\delta &= \mathbb{E}_{c\sim\mathcal{P}_c}[(\delta(c)-\mu_\delta)(\delta(c)-\mu_\delta)^\top] \\ &= J_l B_{\mathrm{ctx}} \Sigma_\zeta B_{\mathrm{ctx}}^\top J_l^\top. \end{aligned} \tag{54}$$

Therefore

$$\mathrm{rank}(\Sigma_\delta) \le \min\{\mathrm{rank}(J_l B_{\mathrm{ctx}}), \mathrm{rank}(\Sigma_\zeta)\} \le \mathrm{rank}(B_{\mathrm{ctx}}) \le r_c. \tag{55}$$

If $q_\delta = 0$, then $d_{\mathrm{eff}} = 0$ by eq. (6). If $q_\delta > 0$, Cauchy–Schwarz applied to the positive eigenvalues $\lambda_1, \dots, \lambda_{q_\delta}$ gives

$$\left(\sum_{j=1}^{q_\delta} \lambda_j\right)^2 \le q_\delta \sum_{j=1}^{q_\delta} \lambda_j^2. \tag{56}$$

Dividing by $\sum_{j=1}^{q_\delta} \lambda_j^2 > 0$ yields

$$d_{\mathrm{eff}} \le q_\delta = \mathrm{rank}(\Sigma_\delta) \le r_c. \tag{57}$$

□

Nonlinear key maps. If g_l is differentiable but not affine, let $J_l = Dg_l(h_l(q_e)) \in \mathbb{R}^{d_m \times d}$ be its Jacobian at the clean hidden state. Taylor's theorem gives

$$\delta(c) = J_l(B_{\mathrm{ctx}}\zeta(c) + b_{\mathrm{ctx}}) + \rho_l(c), \tag{58}$$

where $\rho_l(c) \in \mathbb{R}^{d_m}$ is the Taylor remainder. If the Jacobian is locally L_J-Lipschitz in operator norm for some $L_J > 0$, then

$$\|\rho_l(c)\|_2 \le \frac{L_J}{2}\|B_{\mathrm{ctx}}\zeta(c) + b_{\mathrm{ctx}}\|_2^2. \tag{59}$$

The exact rank bound in theorem 2.4 applies to the affine component. A nonzero remainder may increase the rank of the full shift covariance, so the empirical eigenspectrum should be interpreted as a diagnostic of compactness, not a guarantee of finite-context coverage. Off-support behavior for unseen contextual keys is treated separately by corollary 3.2.

C.5 Derivation of the Wasserstein robust upper bound

Proof of proposition 3.1. By the coupling characterization of the 2-Wasserstein distance (Villani, 2009), for any $\mathcal{Q} \in \mathbb{B}_\epsilon(\mathcal{P}_0)$ and any $\xi > 0$, there exists a coupling π of $X \sim \mathcal{P}_0$ and $Y \sim \mathcal{Q}$ such that

$$\mathbb{E}_\pi\|Y - X\|_2^2 \le (\epsilon + \xi)^2. \tag{60}$$

Because $UY = UX + U(Y - X)$, Minkowski's inequality in $L_2(\pi)$ gives

$$\left(\mathbb{E}_{\mathcal{Q}}\|UY\|_2^2\right)^{1/2} \le \left(\mathbb{E}_{\mathcal{P}_0}\|UX\|_2^2\right)^{1/2} + \left(\mathbb{E}_\pi\|U(Y-X)\|_2^2\right)^{1/2}. \tag{61}$$

The first term satisfies

$$\begin{aligned}\mathbb{E}_{\mathcal{P}_0}\|UX\|_2^2 &= \mathbb{E}_{\mathcal{P}_0}[X^\top U^\top U X] \\ &= \mathrm{Tr}\big(U\,\mathbb{E}_{\mathcal{P}_0}[XX^\top]U^\top\big) \\ &= \mathrm{Tr}(UM_0U^\top) \\ &\le \mathrm{Tr}(UCU^\top), \end{aligned} \tag{62}$$

where $M_0 = \mathbb{E}_{\mathcal{P}_0}[XX^\top]$ and the inequality follows from the main-text assumption $C \succeq M_0$. The second term satisfies

$$\begin{aligned}\mathbb{E}_\pi\|U(Y-X)\|_2^2 &\le \|U\|_{\mathrm{op}}^2 \mathbb{E}_\pi\|Y - X\|_2^2 \\ &\le (\epsilon+\xi)^2\|U\|_{\mathrm{op}}^2. \end{aligned} \tag{63}$$

Combining eqs. (61) to (63),

$$\left(\mathbb{E}_{\mathcal{Q}}\|UY\|_2^2\right)^{1/2} \le \sqrt{\mathrm{Tr}(UCU^\top)} + (\epsilon+\xi)\|U\|_{\mathrm{op}}. \tag{64}$$

For nonnegative a, b and any $\beta > 0$, Young's inequality gives

$$\begin{aligned}(a+b)^2 &= a^2 + 2ab + b^2 \\ &\le a^2 + \beta a^2 + \beta^{-1} b^2 + b^2 \\ &= (1+\beta)a^2 + (1+\beta^{-1})b^2. \end{aligned} \tag{65}$$

Applying this inequality to eq. (64), taking $\xi \downarrow 0$, and using $\|U\|_{\mathrm{op}} \le \|U\|_F$ yields

$$\begin{aligned}\mathcal{R}_\epsilon(U) &\le (1+\beta)\,\mathrm{Tr}(UCU^\top) + (1+\beta^{-1})\epsilon^2 \|U\|_F^2 \\ &= (1+\beta)\left[\mathrm{Tr}(UCU^\top) + \frac{\epsilon^2}{\beta}\,\mathrm{Tr}(UU^\top)\right] \\ &= (1+\beta)\,\mathrm{Tr}\left[U\left(C + \frac{\epsilon^2}{\beta} I_{d_m}\right)U^\top\right]. \end{aligned} \tag{66}$$

The second equality uses

$$\frac{1+\beta^{-1}}{1+\beta} = \frac{(\beta+1)/\beta}{\beta+1} = \frac{1}{\beta}. \tag{67}$$

This proves eq. (13). □

Shift-calibrated covariance inflation. Assume $\epsilon > 0$ and $s_\delta > 0$. Choosing

$$\beta = \frac{\epsilon}{s_\delta}, \quad s_\delta = \sqrt{\mathrm{Tr}(\widehat{\Sigma}_\delta)/d_m} \tag{68}$$

produces

$$\frac{\epsilon^2}{\beta} = \epsilon s_\delta = \epsilon\sqrt{\mathrm{Tr}(\widehat{\Sigma}_\delta)/d_m}. \tag{69}$$

The bound remains valid for every positive β; this choice calibrates the covariance inflation to the root-mean-square coordinate dispersion estimated by eq. (10). It is an empirical calibration rule, not a confidence-region guarantee. If $s_\delta = 0$, the main-text rule gives $\epsilon^* = 0$ and $C_{\epsilon^*} = C$, so no division by s_δ is required.

For $m > 0$, substituting the plug-in effective dimension into the main-text calibration gives

$$\epsilon^* = s_\delta\sqrt{\frac{\widehat{d}_{\mathrm{eff}}}{m}}, \qquad \beta^* = \frac{\epsilon^*}{s_\delta} = \sqrt{\frac{\widehat{d}_{\mathrm{eff}}}{m}}, \qquad \frac{(\epsilon^*)^2}{\beta^*} = s_\delta^2\sqrt{\frac{\widehat{d}_{\mathrm{eff}}}{m}}. \tag{70}$$

Thus ϵ^* has the units of a key displacement, while $(\epsilon^*)^2/\beta^* = \epsilon^* s_\delta$ has the units of the second moment C and yields C_{ϵ^*} in eq. (14). The calibration increases with empirical dispersion and effective dimension and decreases with the number of retained contexts.

C.6 Closed-form solution of the robust multi-key write

Proof of eq. (15). Introduce a Lagrange multiplier $\Lambda \in \mathbb{R}^{d\times(m+1)}$ for the constraint $UK = \Delta V$. The Lagrangian is

$$\mathcal{J}(U, \Lambda) = \mathrm{Tr}(UC_{\epsilon^*}U^\top) + 2\,\mathrm{Tr}\left[\Lambda^\top(UK - \Delta V)\right]. \tag{71}$$

Its derivative with respect to U is

$$\nabla_U \mathcal{J} = 2UC_{\epsilon^*} + 2\Lambda K^\top. \tag{72}$$

The stationarity condition gives

$$\begin{aligned} UC_{\epsilon^*} + \Lambda K^\top &= 0, \\ U &= -\Lambda K^\top C_{\epsilon^*}^{-1}. \end{aligned} \tag{73}$$

Substituting into the constraint,

$$\begin{aligned} UK &= -\Lambda K^\top C_{\epsilon^*}^{-1} K = \Delta V, \\ \Lambda &= -\Delta V (K^\top C_{\epsilon^*}^{-1} K)^{-1}. \end{aligned} \tag{74}$$

Substituting back into eq. (73) yields

$$U^* = \Delta V (K^\top C_{\epsilon^*}^{-1} K)^{-1} K^\top C_{\epsilon^*}^{-1}. \tag{75}$$

Thus $U^* = \Delta W^*$ in eq. (15). Since $C_{\epsilon^*} \succ 0$, the quadratic objective is strictly convex in U. Invertibility of $K^\top C_{\epsilon^*}^{-1} K$ ensures a unique constrained minimizer.

For $m = 0$, write $K = [k]$ and $\Delta V = [v^* - Wk]$. Setting $\epsilon^* = 0$ gives $C_{\epsilon^*} = C$, so

$$\begin{aligned} U^* &= (v^* - Wk)(k^\top C^{-1} k)^{-1} k^\top C^{-1} \\ &= (v^* - Wk) \frac{k^\top C^{-1}}{k^\top C^{-1} k}, \end{aligned} \tag{76}$$

which is exactly the rank-one update in eq. (3). □

Proof of corollary 3.2. Because $C_{\epsilon^*} \succ 0$, its symmetric square root and inverse square root exist. For any $e \in \mathbb{R}^{d_m}$,

$$\begin{aligned} \|U^* e\|_2 &= \left\| U^* C_{\epsilon^*}^{1/2} C_{\epsilon^*}^{-1/2} e \right\|_2 \\ &\le \left\| U^* C_{\epsilon^*}^{1/2} \right\|_F \left\| C_{\epsilon^*}^{-1/2} e \right\|_2 \\ &= \sqrt{\mathrm{Tr}(U^* C_{\epsilon^*} (U^*)^\top)} \sqrt{e^\top C_{\epsilon^*}^{-1} e}. \end{aligned} \tag{77}$$

Squaring both sides proves eq. (17). For $k' = K\alpha + e_k$ and $v' = V^*\alpha + e_v$, feasibility gives

$$\begin{aligned} \widehat{W} k' - v' &= (W + U^*)(K\alpha + e_k) - (V^*\alpha + e_v) \\ &= W e_k - e_v + U^* e_k, \end{aligned} \tag{78}$$

where $(W + U^*)K = V^*$. Applying the triangle inequality and then eq. (17) with $e = e_k$ yields eq. (18). □

Numerical stabilization. When the selected keys are nearly dependent, the implementation uses

$$\Delta W_{\mathrm{ridge}} = \Delta V (K^\top C_{\epsilon^*}^{-1} K + \lambda_{\mathrm{ridge}} I_{m+1})^{-1} K^\top C_{\epsilon^*}^{-1}, \quad \lambda_{\mathrm{ridge}} > 0. \tag{79}$$

Let $\Gamma_K = K^\top C_{\epsilon^*}^{-1} K$. The interpolation residual is

$$\begin{aligned} \Delta W_{\mathrm{ridge}} K - \Delta V &= \Delta V (\Gamma_K + \lambda_{\mathrm{ridge}} I_{m+1})^{-1} \Gamma_K - \Delta V \\ &= \Delta V (\Gamma_K + \lambda_{\mathrm{ridge}} I_{m+1})^{-1} \left[\Gamma_K - (\Gamma_K + \lambda_{\mathrm{ridge}} I_{m+1}) \right] \\ &= -\lambda_{\mathrm{ridge}} \Delta V (\Gamma_K + \lambda_{\mathrm{ridge}} I_{m+1})^{-1}. \end{aligned} \tag{80}$$

Therefore

$$\|\Delta W_{\mathrm{ridge}} K - \Delta V\|_F \le \lambda_{\mathrm{ridge}} \|\Delta V\|_F \|(\Gamma_K + \lambda_{\mathrm{ridge}} I_{m+1})^{-1}\|_{\mathrm{op}}. \tag{81}$$

As $\lambda_{\mathrm{ridge}} \downarrow 0$, the stabilized update converges to eq. (15) whenever Γ_K is nonsingular.

Objective value at the optimum. Substituting eq. (75) into the quadratic objective gives a closed form. Let $\Gamma_K = K^\top C_{\epsilon^*}^{-1} K$. Then

$$\begin{aligned} U^* C_{\epsilon^*} U^{*\top} &= \Delta V \Gamma_K^{-1} K^\top C_{\epsilon^*}^{-1} C_{\epsilon^*} C_{\epsilon^*}^{-1} K \Gamma_K^{-1} \Delta V^\top \\ &= \Delta V \Gamma_K^{-1} (K^\top C_{\epsilon^*}^{-1} K) \Gamma_K^{-1} \Delta V^\top \\ &= \Delta V \Gamma_K^{-1} \Delta V^\top, \end{aligned} \tag{82}$$

and hence

$$\min_{U : UK = \Delta V} \mathrm{Tr}(U C_{\epsilon^*} U^\top) = \mathrm{Tr}\left[\Delta V (K^\top C_{\epsilon^*}^{-1} K)^{-1} \Delta V^\top \right]. \tag{83}$$

This proves eq. (16). The expression makes the coverage–conditioning trade-off explicit: adding a context contributes a new constraint, but nearly dependent contextual keys can make the Gram matrix ill-conditioned and increase the energy required to satisfy all targets.

L1: Natural support for the pre-edit answer

Peer-message generation prompt

You are the {role} in a simulated peer discussion.
Question: {query}
Recorded answer: {old}
Express the recorded answer naturally in one short factual statement.
Do not answer a different question.
Peer statement:

Controlled upstream roles

worker_1

Message delivered to Worker 2

Recorded answer: {old}. {natural_statement}

Edited-worker input

{worker_2_role_instruction}
[worker_1]: {message}
Question: {query}
Your logical analysis:

L2: Explicit insistence on retaining the pre-edit answer

Peer-message generation prompt

You are the {role} in a simulated peer discussion.
Question: {query}
Recorded answer: {old}
Express the recorded answer naturally in one short factual statement.
Do not answer a different question.
Peer statement:

Controlled upstream roles

worker_1

Message delivered to Worker 2

Recorded answer: {old}. My colleague insists that the answer is {old}.
{natural_statement}

Edited-worker input

{worker_2_role_instruction}
[worker_1]: {message}
Question: {query}
Your logical analysis:

L3: Consistent support from both peers

Peer-message generation prompt

You are the {role} in a simulated peer discussion.
Question: {query}
Recorded answer: {old}
Express the recorded answer naturally in one short factual statement.
Do not answer a different question.
Peer statement:

Controlled upstream roles

manager and worker_1

Message delivered to Worker 2

Recorded answer: {old}. {natural_statement_for_role}

Edited-worker input

{worker_2_role_instruction}
[manager]: {manager_message}
[worker_1]: {worker_1_message}
Question: {query}
Your logical analysis:

L4: Supporting arguments and rejection of alternatives

Peer-message generation prompt

You are the {role} in a simulated peer discussion.
Question: {query}
Recorded answer: {old}
Defend the recorded answer with a short supporting argument
and a reason to reject alternatives. Do not answer a different question.
Peer statement:

Controlled upstream roles

manager and worker_1

Message delivered to Worker 2

Recorded answer: {old}. I defend {old} and reject conflicting answers.
Supporting argument: {supporting_argument_for_role}

Edited-worker input

{worker_2_role_instruction}
[manager]: {manager_message}
[worker_1]: {worker_1_message}
Question: {query}
Your logical analysis:

Figure 9: Peer-pressure prompt templates for L1–L4.

D ALL PROMPTS

The following contents present the role instructions and input templates used in the main evaluation.

D.1 L1–L4 PROMPTS

Figure 9 shows the peer-message prompts used in the pressure experiment in Section B.7. The four levels progress from natural support for the pre-edit answer to explicit insistence, support from two peers, and arguments that reject alternatives. This progression varies the strength and number of opposing peer messages while keeping the factual question and edited role fixed.

D.2 ARCHITECTURE PROMPTS

Figures 10, 11, and 12 show the role instructions and appended input templates for the three multi-agent architectures. The templates specify which preceding agent messages each role receives; the placeholders are filled with the corresponding question and messages at runtime. They also make explicit how the edited role's input depends on the communication history in each architecture.

MAKE Pipeline: Planner

Role instruction (verbatim)

You are the Planner. Given a knowledge question, your job is to DECOMPOSE it into 2-4 sub-questions that a factual executor will answer.
IMPORTANT: You MUST output the sub-questions now — never leave your response empty.
Write each sub-question on its own line, as a plain question (e.g. 'What was X? What did they make?').

Appended input

Question: {query}

Decompose into sub-questions:

MAKE Pipeline: Executor

Role instruction (verbatim)

You are the Executor in a multi-agent pipeline. The Planner has decomposed a question. Answer with PRECISE factual information.
CRITICAL: The Planner may have OUTDATED knowledge. You have CURRENT information. If your knowledge contradicts the Planner, output YOUR correct answer — do NOT defer.
FORMAT: Start your response with a single line exactly like:
ANSWER: <your one-sentence factual answer>
Then optionally add supporting details on subsequent lines.
Example:
ANSWER: The capital of France is Paris.
It is one of the largest cities in Europe.
Always produce a complete answer; never leave your response blank.

Appended input

[Planner]: {planner_response}
Question: {query}

Your answer:

MAKE Pipeline: Reviewer

Role instruction (verbatim)

You are the Reviewer in a multi-agent pipeline.
Your job is to verify whether the Executor's answer is factually correct based on YOUR OWN knowledge, and give the FINAL answer.

RULES (single-round evaluation — no revision rounds):
1. If the Executor's answer is CORRECT and matches what you know: output ONLY the verified answer itself (no commentary).
2. If the answer is WRONG, INCOMPLETE, or contradicts your knowledge: output the CORRECT answer as you know it. Do NOT write 'UNCERTAIN' and do NOT add explanation or commentary.
3. If the answer has no factual content or is gibberish: output the correct answer from your own knowledge.
4. ALWAYS output a single definitive answer; never leave your output blank.
5. Do NOT repeat or copy the question, the plan, or the Executor's answer — output only your own review.

Example:
Question: The Eiffel Tower is located in
[Executor]: ANSWER: The Eiffel Tower is located in Paris.
Your review: The Eiffel Tower is located in Paris.

Appended input

Question: {query}
[Planner]: {planner_response}
[Executor]: {executor_response}

Your review:

Figure 10: Role prompts for the Pipeline architecture.

MAKE Debate: Debater A

Role instruction (verbatim)

You are Debater A, a knowledgeable expert. Your task is to STATE what you know about the given topic with confidence and precision. Provide specific facts, details (names, dates, places), and clear reasoning supported by your knowledge. Do NOT hedge or speculate — speak as an authority. Always provide a complete response; never leave your answer blank.

Appended input

Question: {query}

Your argument:

MAKE Debate: Debater B

Role instruction (verbatim)

You are Debater B, a knowledgeable expert with access to CURRENT information. Review Debater A's statement and provide YOUR OWN independent factual answer to the question.
IMPORTANT: Debater A's information may be OUTDATED. You have been updated with the latest factual knowledge. If your knowledge contradicts Debater A's claim, state YOUR correct answer confidently with specific supporting details — do NOT agree with outdated claims.
— If you DISAGREE with Debater A: explain why with specific facts.
— If you AGREE: provide the facts in your own words with additional supporting details.
In either case, ALWAYS produce a complete factual response. Never leave your response blank.

Appended input

[debater_a]: {debater_a_response}

Question: {query}

Your counter-argument:

MAKE Debate: Judge

Role instruction (verbatim)

You are the Judge. Two debaters have presented factual claims. Your task is to determine the CORRECT factual answer.
RULES (single-round evaluation — no revision rounds):
1. Evaluate both claims based on your own knowledge. One debater may have outdated info — trust the facts, not seniority.
2. If you are confident: output ONLY the answer itself (no labels, no commentary).
3. If you cannot determine the answer with confidence: output the answer you believe is correct based on your own knowledge. Do NOT write 'UNCERTAIN' and do NOT add commentary.
4. If the debaters' answers have no factual content or are gibberish: output the correct answer from your own knowledge.
5. ALWAYS output a single definitive answer; never leave your output blank.
6. Do NOT repeat or copy the question or the debaters' statements — output only your own judgment.

Example:
Question: The Eiffel Tower is located in
[debater_a]: The Eiffel Tower is located in Berlin.
[debater_b]: The Eiffel Tower is located in Paris.
Your judgment: The Eiffel Tower is located in Paris.

Appended input

Question: {query}
[debater_a]: {debater_a_response}
[debater_b]: {debater_b_response}

Your judgment:

Figure 11: Role prompts for the Debate architecture.

MAKE Hierarchical: Manager (delegation)

Role instruction (verbatim)

You are the Manager. Given a factual question, delegate it to two specialized workers by describing what each should investigate:
— Worker 1 (factual knowledge): assign a specific factual question.
— Worker 2 (logical reasoning): assign a specific logical analysis task.
IMPORTANT: Only describe the task assignments. Do NOT simulate or generate the workers' responses yourself — the actual workers will produce those.

When synthesizing the workers' outputs into a final answer:
1. PREFER Worker 1's factual answer — Worker 1 has specialized knowledge that may be more current than yours.
2. Use Worker 2's analysis for consistency, not for overriding facts.
3. Output only the factual answer itself — no prefix, label, or header. If workers' outputs are genuinely contradictory or unusable, output the answer you believe is correct based on your own knowledge. Do NOT write 'UNCERTAIN'.

Appended input

Question: {query}

Delegate the question to workers:

MAKE Hierarchical: Worker 1

Role instruction (verbatim)

You are Worker 1, a factual knowledge expert. Answer the Manager's delegated question with PRECISE, detailed factual information. State specific names, dates, locations, and evidence from your knowledge. Be definitive and confident — do NOT hedge. Always provide a complete factual response; never leave your answer blank.

Appended input

Question: {query}
[Manager]: {manager_response}

Your factual answer:

MAKE Hierarchical: Worker 2

Role instruction (verbatim)

You are Worker 2, a reasoning and logic expert. Analyze the delegated question from a logical perspective. For any question — even simple ones — provide logical analysis such as: verifying factual consistency, identifying underlying assumptions, checking for logical fallacies, explaining the reasoning chain, or confirming the conclusion follows from the premises. Be precise and rigorous. Always provide a complete analysis; never leave your response blank.

Appended input

[Manager]: {manager_response}

Question: {query}

Your logical analysis:

MAKE Hierarchical: Manager (synthesis)

Role instruction (verbatim)

You are the Manager synthesizing the FINAL answer. Combine the two workers' outputs into a single definitive answer.

RULES (single-round synthesis — no revision rounds):
1. If Worker 2's answer is CORRECT and matches what you know: output ONLY the verified answer itself (no commentary).
2. If Worker 2's answer is WRONG, INCOMPLETE, or contradicts your knowledge: output the CORRECT answer as you know it. Do NOT write 'UNCERTAIN'.
3. Use Worker 1's factual answer as supporting evidence, not for overriding the judgment.
4. If the workers' outputs are contradictory or unusable: output the answer you believe is correct based on your own knowledge.
5. ALWAYS output a single definitive answer; never leave your output blank.
6. Do NOT repeat or copy the question or the workers' statements — output only your own synthesis.

Example:
Question: The Eiffel Tower is located in
[worker_1]: The Eiffel Tower is located in Berlin.
[worker_2]: The Eiffel Tower is located in Paris.
Your synthesis: The Eiffel Tower is located in Paris.

Appended input

Question: {query}
[worker_1]: {worker_1_response}
[worker_2]: {worker_2_response}

Synthesize the final answer:

Figure 12: Role prompts for the Hierarchical architecture.