PREMO: COMPLEMENTARY TRAINING FOR PERSONALIZED PREFERENCE DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Personalized decisions require discovering query-relevant user preferences from long, noisy histories. These preferences can be summarized in a memo that covers relevant preferences while remaining faithful to historical evidence. However, the complete set of relevant preferences is unobserved, making memo discovery difficult to supervise. We introduce **PREMO** (**PRE**ference **M**em**O**), which alternates a single model between two complementary roles. As a proposer, it discovers new atomic preferences and is rewarded by independent verification against the history. As a memo policy, it generates a memo from the query and history alone, rewarded for covering verified preferences without asserting rejected ones. Proposer training expands the verified supervision and directly trains preference recognition, while memo training learns to express these preferences in a useful summary. Our theoretical analysis identifies conditions under which proposer training followed by memo training improves both coverage and faithfulness over memo-only training. We evaluate PREMO on shopping, movie recommendation, and everyday decisions. A trained 4B model writes memos of about 170 tokens that effectively summarize histories 40 to 100 times longer. It improves coverage and faithfulness over memo-only training and raises downstream accuracy by up to 3.7 points. Its memos also match or beat its frontier-model teacher's own: PREMO covers more and raises downstream accuracy by up to 6.8 points, even when GPT-5.4 itself decides. The advantage over memo-only training holds at 35B.

1 INTRODUCTION

Large Language Model (LLM) assistants are increasingly deployed to help users write documents, choose products, find movies, and make everyday plans (Dai et al., 2024; Yang et al., 2026; Jiang et al., 2026). The usefulness of those *personalized* decisions depends on how well they match the tastes and needs of the person asking (Chen et al., 2025). It requires the agent to discover user preferences associated with the request from noisy user histories (Koren, 2009; Wang et al., 2021; Zhao et al., 2025). For example, to help a user furnish their living room, it should infer the taste from the Bauhaus chairs and hotel lobby they saved, not from a plush toy bought for a niece (Figure 1). Such clues and distractors accumulate across browsing records, purchases, and conversations. Personalization therefore requires discovering which preferences in this history should guide the current request, without treating every past action as evidence of what the user wants now.

We study the preference discovery problem through a *memo*: a compact, query-conditioned account of user preferences that a downstream policy can use directly and needs to satisfy two requirements: (i) *coverage* of supported, query-relevant preferences (Zhao et al., 2025; Yang et al., 2026), and (ii) *faithfulness* to the evidence, avoiding unsupported, contradicted, or inapplicable claims (Hu et al., 2026; Sun et al., 2026). However, it lacks *direct supervision* to discover reliable memos: the complete set of relevant preferences is unobserved, and even a known downstream choice does not reveal every preference that should inform it. Hence, it raises a challenging research question:

How to train LLM agents to discover reliable memos from long and noisy user histories?

Existing approaches address these two requirements *only partially*. Methods based on raw or retrieved histories (Zhao et al., 2025) and external memories (Chhikara et al., 2025) make user information accessible, but relevant evidence can still be overlooked in long contexts (Liu et al., 2024).

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.

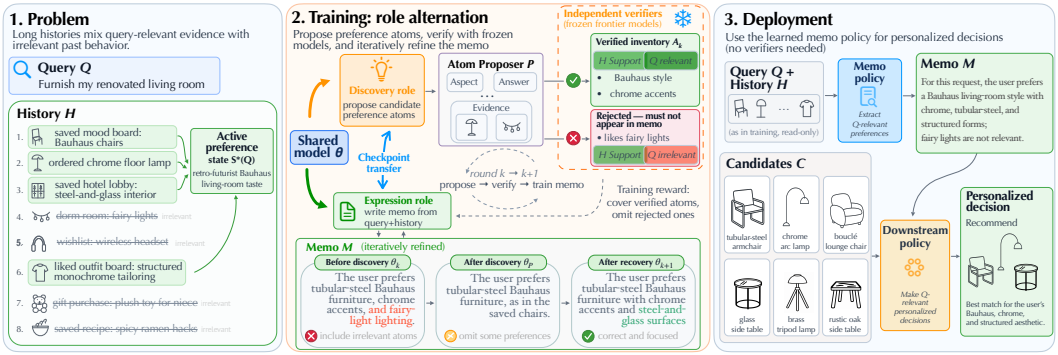


Figure 1: Overview of PREMO. Left: the task, a query and a history that mixes relevant and irrelevant records. Middle: one model in two roles: discovery (the proposer role) proposes atoms that frozen verifiers accept or reject; expression (the memo policy) writes the memo from query and history, rewarded for covering verified atoms and omitting rejected ones; bottom: memos at θ_k, θ_P, and θ_{k+1}. Right: at deployment the downstream policy reads the memo, replacing the raw history.

Learned preference summaries and personalized evaluation criteria can improve downstream decisions (Chen et al., 2025; Seo & Lee, 2026), yet usefulness does not imply faithfulness. Personalized systems remain vulnerable to unsupported profile inferences (Sun et al., 2026) and identity-based generalizations (Kantharuban et al., 2024). Using an LLM judge does not fully resolve this problem, as assessing correctness against contextual evidence remains challenging (Xu et al., 2025a). Conversely, claim-level verification can assess the evidential support of individual statements (Wei et al., 2024). However, verification only checks the preferences that are stated; it gives no reward for discovering new ones. The resulting memos may therefore be *informative but insufficiently grounded*, or *faithful but incomplete*, limiting their reliability for personalized decisions.

To address this challenge, we formulate memo discovery as recovering a latent set of query-relevant preference atoms (Sec. 2). Our formulation links memo coverage and faithfulness to two capabilities: *recognition* of history-supported, query-relevant preferences and their *expression* in the memo. A model may recognize a preference yet fail to express it (Burns et al., 2023). In an idealized model of coupled learning dynamics (Saxe et al., 2014), optimizing only memo quality leads to a *recognition–expression bottleneck*, in which the two capabilities slow each other's improvement when both are weak. We establish conditions under which directly training recognition before optimizing memo quality reduces both risks relative to optimizing memo quality alone at a matched budget, demonstrating the complementarity of these two objectives (Proposition 3; Theorem A.4 under GRPO updates).

Guided by this, we introduce **PREMO** (**PRE**ference **M**em**O**), which alternates a single shared model between two complementary roles (Figure 1). As a *proposer*, the model reads the query, history, and current inventory, and proposes new preference atoms with cited evidence (Section 3.2). Independent frozen verifiers check the proposals against the history, rewarding the proposer and adding accepted atoms to the inventory (Section 3.1). As a *memo policy*, the same model generates a memo from the query and history alone, rewarded for covering verified preferences without asserting invalid claims (Section 3.3). The proposer explicitly trains preference recognition, while training the memo policy requires both recognition and expression. Training the proposer supplies both new verified preferences and an updated shared checkpoint for the next memo policy training. Repeated alternation expands the inventory and trains the model to express it in a useful memo (Section 3.4). At deployment, the memo policy writes the memo in one call for the downstream policy.

We evaluate PREMO on personalized recommendation (APeB (Yang et al., 2026) and PersonaMovieLens, built on MovieLens (Harper & Konstan, 2015)) and on the daily-life tasks of PersonaMem-v3 (Jiang et al., 2026). A trained Qwen3.5-4B writes memos of about 170 tokens. They beat the histories they replace, 40 to 100 times longer. It also beats memo-only training at matched updates in coverage and faithfulness, and by up to 3.7 points in downstream accuracy. Its memos cover more than those of GPT-5.4, its frontier-model teacher, and match or beat them downstream, by up to 6.8 points even when GPT-5.4 itself decides. The gains persist at 35B.

2 THEORY: WHY COMPLEMENTARY ROLES HELP PREFERENCE LEARNING

In this section, we formalize the discovery problem and begin by formally defining the memo.

Definition 1 (Preference memo)**.** Given a query Q and history H, a *preference memo* M is a compact, query-conditioned account of the preferences H supports.

A *memo policy* generates $M \sim p_\theta^M(\cdot \mid Q, H)$. For downstream prediction, let C denote the candidate set and Y the target choice or response. The downstream policy produces $\widehat{Y} \sim \pi_\psi(\cdot \mid Q, M, C)$, which is evaluated against Y. A memo discards information, but a downstream policy that reads a short memo can carry a smaller excess risk than one reasoning over the full history; Appendix B.3 bounds the decision-risk difference, which RQ1 tests.

What makes a good memo? A reliable memo should cover the preferences supported by the history and relevant to the query (*coverage*), while avoiding unsupported, contradicted, or inapplicable claims (*faithfulness*). We represent preference claims as *atoms*, each concerning one preference aspect and accompanied by cited historical records. For $x = (Q, H)$, let us denote the set of supported, query-relevant atoms as $\mathcal{F}^+(x)$ and the set of invalid claims as $S^-(x)$.

Definition 2 (Coverage and faithfulness risks)**.** Let $m_\theta(r \mid x)$ be the probability that the memo expresses r, $a_{x,r} \geq 0$ be the weight of atom r (set by verification, Section 3.1), and $B \subseteq \mathcal{F}^+$, define

$$C_M^B(\theta) = \mathbb{E}_x \sum_{r \in B(x)} a_{x,r}[1 - m_\theta(r \mid x)], \quad H_M(\theta) = \mathbb{E}_x \sum_{r \in S^-(x)} a_{x,r} m_\theta(r \mid x). \tag{1}$$

they jointly penalize omissions and invalid claims, respectively via $R_M = C_M^{\mathcal{F}^+} + \lambda H_M$, $\lambda > 0$.

To analyze R_M, we distinguish *recognition*, measured by the probability $p_\theta(r \mid x)$ that the model judges a supplied atom valid when asked directly, from *expression*, measured by $m_\theta(r \mid x)$. Now let $B \subseteq \mathcal{F}^+$ be a subset of the valid atoms, such as those discovered so far. Then, let Ψ_B^+ and Ψ^- be the expected weighted sums of $1 - \min(p_\theta, m_\theta)$ over B and $\max(p_\theta, m_\theta)$ over S^-, we have $C_M^B \leq \Psi_B^+$ and $H_M \leq \Psi^-$.

Proposition 1 (Memo risk decomposition)**.** *Let $U_B = \mathbb{E}_x \sum_{r \in \mathcal{F}^+(x) \setminus B(x)} a_{x,r}$ denote the valid-atom weight outside B, we have $R_M(\theta) \leq U_B + \Psi_B^+(\theta) + \lambda \Psi^-(\theta)$.*

The proof is given in Appendix A.1. We next ask whether separately training recognition can improve subsequent memo optimization under a fixed training budget.

Theorem 2 (Recognition–expression bottleneck)**.** *Consider scalar recognition and expression factors u, v, with memo gain $g = uv$ and loss $L_M = \frac{1}{2}(1 - uv)^2$. Memo training follows gradient flow $\dot{u} = v(1 - uv), \quad \dot{v} = u(1 - uv)$. Starting from $u_0 = v_0 = \sigma \in (0, 1)$, its gain satisfies*

$$g_D(t) = \frac{1}{1 + (\sigma^{-2} - 1)e^{-2t}}. \tag{2}$$

As $\sigma \to 0$, reaching any fixed target gain in $(0, 1)$ requires $\log(1/\sigma) + O(1)$ training time.

The proof is in Appendix A.2. Theorem 2 quantifies the **recognition–expression bottleneck**: each factor's update is scaled by the other, so weak recognition and expression slow each other.

Complementary training. This bottleneck motivates directly strengthening recognition before optimizing the memo. Consider an auxiliary recognition loss $L_P = \frac{1}{2}(1 - u)^2$, whose gradient flow is $\dot{u} = 1 - u$, $\dot{v} = 0$. Recognition can therefore improve without depending on expression. The next result establishes when this benefit outweighs the time allocated away from memo training.

Proposition 3 (Complementary training)**.** *Under the model of Theorem 2, start from $u_0 = v_0 = \sigma$ and train on L_P for $t_P > 0$, followed by L_M for $t_M > 0$. Let g_A denote the final gain and $T = t_P + t_M$ the total training duration. Define $\alpha = 1 - (1 - \sigma)e^{-t_P}$ and $c = \alpha^2 - \sigma^2$. Then*

$$g_A \geq \underline{g}_A = 1 - (1 - \alpha\sigma)e^{-ct_M}, \qquad \Delta_{\mathrm{bn}} := \underline{g}_A - g_D(T). \tag{3}$$

For every fixed $t_P, t_M > 0$, there exists $\sigma_0 > 0$ such that $\Delta_{\mathrm{bn}} > 0$ for all $\sigma \in (0, \sigma_0)$. Thus the two-stage schedule achieves strictly higher gain and lower memo loss than memo-only training.

The proof is in Appendix A.2. Recognition training raises u to α, which stays positive as $\sigma \to 0$, so memo training can then improve expression with stronger recognition, alleviating the bottleneck.

To connect the gain to Eq. 1, additionally assume that memo expression probabilities on feasible valid and invalid atom subsets are $(1+g)/2$ and $(1-g)/2$, respectively. For subsets of total weights W_C and W_H, the corresponding coverage and faithfulness risks are $W_C(1-g)/2$ and $W_H(1-g)/2$. Proposition 3 therefore lowers each risk by at least $W_j\Delta_{\mathrm{bn}}/2$, $j \in \{C, H\}$ (Appendix A.5). Sufficient conditions for this advantage to persist under actual GRPO updates are given in Appendix A.4.

3 The PREMO Framework

In this section, we propose PREMO to realize the complementary training principle of Prop. 3 with a shared model acting as a *proposer* and a *memo policy*. Both roles require supervision about which preferences the history supports. We first describe how independent verification converts proposed preference atoms into this supervision (Section 3.1). The verification outcomes reward proposer outputs (Section 3.2), while the accumulated accepted and rejected claims provide reference targets for memo evaluation (Section 3.3). Finally, we alternate the roles so that proposer training supplies both an updated checkpoint and additional verified supervision for memo optimization (Section 3.4).

3.1 Constructing Supervision through Verification

Verification operates on the atoms of Section 2. Each proposal specifies a preference question, its answer, and supporting records cited from the pre-query history. A verification path $\mathcal{V}$ combines rule-based grounding with judgments by frozen frontier models independent of the trained policy. Splitting the checks makes claims easier to judge against long histories (Xu et al., 2025a).

For each instance, the inventory A_k contains atoms accepted before round k; newly accepted, non-duplicate atoms form N_k, and task-invalid or refuted claims accumulate in an observed rejection pool $\widehat{S}_k^-$. Figure 2 summarizes the steps; details are in Appendix C.10.

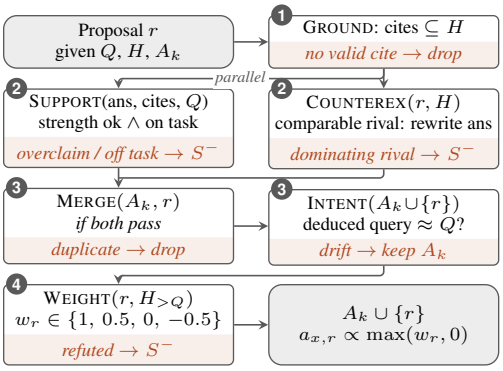


Figure 2: Verification path $\mathcal{V}$ for one proposal. Tinted strips: outcome when a check fails.

❶ **Grounding.** A rule check drops proposals whose cited records are missing from H.

❷ **Support and counterexample checks.** In parallel, one checker asks whether the cited records support the preference, another searches H for rivals (comparable ones rewrite the answer, dominating ones reject the atom); rejected atoms go to S^-.

❸ **Merge and consistency.** Duplicates of atoms in A_k are dropped; if the query deduced from the valid atoms has drifted, the round keeps A_k.

❹ **Post-query weight.** Each atom is weighted by the user's post-query behavior (full, half, zero if inconclusive, or negative if refuted, which also sends it to S^-). When no post-query behavior exists, the weight is set by how strongly the cited records support the atom.

3.2 Proposer Role: Discovering Supported Preferences

The proposer reads (Q, H, A_k) and generates up to $N_{\max}$ ordered preference atoms. The inventory steers it toward uncovered preferences, and verification tells it whether each claim is supported and applies to Q. This serves as an auxiliary recognition signal that needs no complete memo.

Proposer reward. Each slot receives score s_i: 1 for a new atom passing all checks, 0.5 for an atom passing support and counterexample checks but merged as a duplicate, -0.5 for verifier rejection, -1 for citing a missing record, and 0 for an empty slot. The response reward is

$$R_p = \mathrm{clip}_{[-1,1]}\left[\frac{\sum_{i=1}^{N_{\max}} \gamma^{i-1} s_i}{\sum_{j=1}^{N_{\max}} \gamma^{j-1}} - \tfrac{1}{2}\mathbf{1}[\mathrm{drift}]\right], \tag{4}$$

where $0 \leq \gamma \leq 1$ discounts later slots and drift denotes failure of the query-consistency check. The fixed-slot normalization and rejection penalties favor supported proposals over unsupported additions. We use R_p for response-level GRPO (Shao et al., 2024) (Appendix C.11).

3.3 Memo Policy: Expressing Supported Preferences

The same model generates $M \sim p_\theta^M(\cdot \mid Q, H)$. The inventory and verifiers supply training feedback but are never inputs to the memo policy. The model must therefore infer the relevant preferences from the history and express them in the memo, exercising both recognition and expression.

Memo reward. Let $I_k = \{r \in A_k \cup N_k : w_r > 0\}$ be the positive-weight reference atoms. A frozen judge scores how fully the memo states each atom in $I_k \cup \widehat{S}_k^-$ using $h_r(M) \in \{0, 0.5, 1\}$, corresponding to absent, partial, or full expression. Define

$$\mathrm{cov}(M) = \frac{\sum_{r \in I_k} w_r h_r(M)}{\sum_{r \in I_k} w_r}, \qquad \mathrm{viol}(M) = \frac{\sum_{r \in \widehat{S}_k^-} h_r(M)}{|\widehat{S}_k^-|}.$$

An ideal memo has $\mathrm{cov} = 1$, when $\{r \in I_k : h_r(M) < 1\} = \varnothing$, and $\mathrm{viol} = 0$, when $\{r \in \widehat{S}_k^- : h_r(M) > 0\} = \varnothing$. The memo reward is

$$R_m(M) = \mathrm{clip}_{[-1,1]}[\mathrm{cov}(M) - \lambda\, \mathrm{viol}(M)] + \rho_{\mathrm{fmt}}(M), \tag{5}$$

where $\lambda = 1$ in our experiments and ρ_{fmt} rewards output validity, length and formatting (Appendix C.11). These scores provide finite-set proxies for the coverage and faithfulness criteria in Eq. 1; claims outside both observed sets receive no atom-level score. The memo policy is trained with response-level GRPO using R_m.

3.4 Alternating Training and Deployment

Training begins with a base checkpoint θ_0 and an inventory A_0 proposed and verified by frontier models. Each round trains the proposer, verifies proposals from the resulting checkpoint to collect N_k, and continues training the same model as a memo policy:

$$\begin{aligned} \theta_P &\leftarrow \mathrm{Train}_P(\theta_k; A_k), \qquad N_k \leftarrow \mathcal{V}(A_k, \theta_P), \\ A_{k+1} &= A_k \cup N_k, \qquad\qquad \theta_{k+1} = \theta_{PM} \leftarrow \mathrm{Train}_M(\theta_P; A_{k+1}, \widehat{S}_k^-). \end{aligned} \tag{6}$$

The set arguments to Train_M specify reward supervision. Both roles optimize the shared model parameters; verifiers and the memo judge remain frozen, and each training phase uses a fixed inventory. The handoff transfers both what the model has learned through proposer training and the additional preference targets collected through verification.

Before the first round, the base model is warmed up by SFT on teacher-written memos and then on teacher-written proposals; later rounds use only GRPO. At each handoff, the same checkpoint is evaluated under both role prompts (Figure 3). The memo-only baseline starts from the same checkpoint and uses the same verified supervision with matched parameter-update counts (Appendix C.12).

At deployment, the memo policy generates a memo directly from (Q, H) in one generation call. The downstream policy uses (Q, M, C), where C is the candidate set, without accessing the raw history. Verifiers and the inventory serve training only.

4 Experiments

Our experiments answer four questions. **RQ1**: does PREMO train a memo policy that raises coverage without losing faithfulness, and do its memos improve personalized decisions over the history itself, user summaries, and frontier-written memos? **RQ2**: what are the drawbacks of training on the memo reward alone, and does the proposer role, supervised through verification, complement it (Proposition 3)? **RQ3**: does each round discover new preferences, and does the memo policy learn to express them, raising coverage where a fixed inventory saturates (Proposition 1)? **RQ4**: what does each step of the verification path of Section 3.1 contribute to PREMO over a single judge?

Table 1: Main results with Qwen3.5-4B as both memo policy and downstream policy. Cov., Faith., and Ground. are coverage, faithfulness, and grounding (Section 4.1); Task is the 0–100 task completion score (Appendix C.1); "–": not applied; best per column in bold.

		APeB				PersonaMovieLens				PersonaMem			
Family	Method	Hit@1	Cov.	Faith.	Ground.	Hit@1	Cov.	Faith.	Ground.	Task	Cov.	Faith.	Ground.
History	Query only	55.1	–	–	–	62.4	–	–	–	65.5	–	–	–
	Full history	54.9	–	–	–	64.9	–	–	–	63.5	–	–	–
	Retrieved	55.3	–	–	–	66.1	–	–	–	62.0	–	–	–
Summary	Persona	61.2	14.1	67.1	65.9	72.7	25.3	57.3	65.4	64.5	36.4	24.8	67.7
	Mem0	57.1	43.8	84.5	80.6	65.0	47.1	**89.4**	83.2	75.3	52.2	39.7	**83.7**
	P-Check	61.4	21.2	**87.7**	**88.6**	70.9	50.2	84.7	**88.5**	73.8	51.2	80.8	38.5
Memo	Base	57.8	30.7	74.5	74.9	71.5	66.8	60.0	63.1	61.5	51.2	28.0	55.6
	SFT, GPT-5.4	60.6	61.7	76.1	78.2	70.7	79.2	84.2	85.0	81.0	65.0	76.0	62.1
	Direct memo D	62.2	70.8	71.6	68.3	72.8	84.0	82.3	81.2	81.2	74.9	84.0	62.9
	PREMO	**65.9**	**73.8**	77.1	74.7	**73.7**	**86.8**	84.5	83.5	**83.3**	**82.9**	**89.5**	65.5
Teacher	GPT-5.4 memo[†]	61.5	64.7	88.8	89.6	73.6	82.0	88.6	89.4	82.5	75.3	94.5	67.0

[†]Teacher: GPT-5.4 wrote the SFT memos and proposed the evaluation atoms $E(x)$ (Appendix C.2).

4.1 Setup

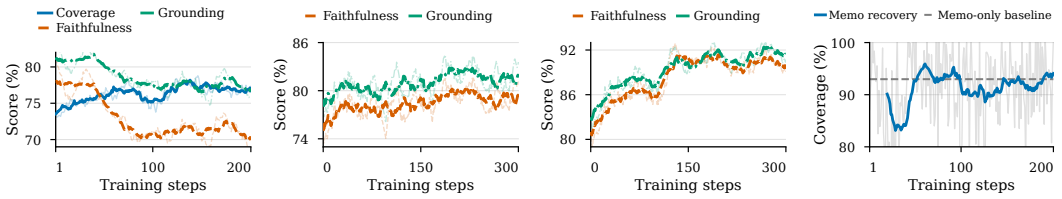
Tasks. Each instance consists of (Q, H, C, Y): query, history, candidates, and target. We evaluate on three personalization tasks. **APeB** (Yang et al., 2026): pick the bought product among five hard candidates, from shopping and video histories. **PersonaMovieLens**: pick the movie the user later rated five stars among five, on single-domain MovieLens 32M (Harper & Konstan, 2015) rating histories with synthesized queries. **PersonaMem-v3** (Jiang et al., 2026): write a personalized reply, without candidates, for 1,251 preference queries over cross-platform histories. Histories average over 10k tokens (≈120 records); users are disjoint across splits (Table 5, Appendix C.1).

Evaluation. Downstream quality is Hit@1 (APeB, PersonaMovieLens) or a 0–100 task completion score on PersonaMem-v3, following the benchmark's criteria of task completion, preference alignment, and rule compliance (Appendix C.1). We further score memos against $E(x)$, the atoms GPT-5.4 proposed and the verification path accepted over four rounds (a verified reference pool, audited in Appendix C.3): *coverage*, the weighted share of $E(x)$ the memo states; *faithfulness*, one minus a 0–1 score for unsupported or overconfident statements; and *grounding*, the share of history citations that support the preference. Numbers average three seeds; judge–human agreement is 95–98% (Appendix C.2); bootstrap intervals for Tables 1 and 2 are in Appendix C.4.

Models and baselines. Qwen3.5-4B is the shared model and, unless stated otherwise, the downstream policy; Section 4.3 adds Qwen3.5-35B-A3B. A_0 comes from one GPT-5.4 round; training runs four rounds of Eq. 6 at 4B and two at 35B, stopping earlier if validation coverage stops rising, with GPT-5.6 Luna as verifier and judge (Appendices C.10 and C.12). Baselines: *history in context* (the query alone, the full history, or the top-30 relevant records ranked by BM25); *user summaries* (a persona, the general preference and checklist of P-Check (Seo & Lee, 2026), and Mem0 (Chhikara et al., 2025), all by Qwen3.5-4B; Appendix C.2); *memo policies* on Qwen3.5-4B (the untrained base, SFT on GPT-5.4 memos, the first stage of the warm-up, and *direct memo training* D, the memo role alone on the same inventories at matched GRPO updates); and, as a reference, memos written by GPT-5.4 itself.

4.2 RQ1: Does PREMO Improve Memos and Decisions (Tables 1 and 2)?

Memos beat the history they compress. PREMO generates a memo of about 170 tokens to replace a history 40 to 100 times longer in the downstream task. With the same downstream policy, decisions from the memo are better on every task (Table 1). Where histories span platforms and domains (APeB, PersonaMem-v3), the history itself is not even guaranteed to beat the query alone. An untrained Base memo already beats the history on APeB, and training widens the margin on every task. Appendices B.3 and C.5 additionally analyze this downstream excess risk.



(a) Memo 4B (b) Proposer 4B (c) Proposer 35B (d) Recovery 35B

Figure 3: Training-prompt curves of single GRPO phases (Appendix C.8). (a) Memo GRPO after warm-up (D) raises coverage and lowers faithfulness and grounding; (b, c) proposer GRPO after warm-up (A) raises both. (d) From the proposer checkpoint θ_P, coverage quickly overtakes D.

Coverage rises without losing faithfulness. Training raises coverage by 20 to 43 points over the untrained Base memo, and faithfulness rises too. The user summaries make the opposite bargain. On APeB and PersonaMovieLens, Mem0 and P-Check are nearly as faithful as GPT-5.4, but Mem0 keeps only clearly relevant records and P-Check writes a general preference without the query. They cover at most half of the query's atoms, and the omissions cost downstream accuracy. For homogeneous movie histories, faithfulness is easier and coverage separates the methods. On PersonaMem, whose history spans platforms and domains, the summaries lose even faithfulness (Mem0) or grounding (P-Check), whereas PREMO stays faithful and leads downstream.

The 4B policy matches its teacher downstream and exceeds it in coverage. GPT-5.4 wrote the SFT memos and proposed the initial inventory A_0. SFT on its memos lifts coverage, and PREMO lifts it further, past the teacher (Appendix C.6). Downstream, the trained 4B memos also match or beat the teacher's on every task, by 4.4 points on APeB, and with most downstream policies (Table 2). Faithfulness is different: SFT does not reproduce the teacher's faithfulness, suggesting that model scale contributes to the gap. The 35B-A3B model trained with PREMO narrows it while keeping the coverage gain (RQ2).

Table 2: Memos by GPT-5.4 and by PREMO under five downstream policies; the better APeB memo per row in bold.

	APeB Hit@1		PersonaMem	
Downstream	GPT-5.4	PREMO	GPT-5.4	PREMO
Qwen3.5-4B	61.5	**65.9**	82.5	83.3
Gemini 3.6 Flash	67.4	**72.8**	91.1	91.0
GPT-5.4	66.6	**73.4**	92.0	92.7
GPT-5.5	69.4	**72.4**	92.8	92.5
GPT-5.6 Luna	67.0	**70.2**	91.8	92.0

4.3 RQ2: What Do the Two Roles Each Contribute (Table 3, Figure 3)?

Route A alternates proposer and memo phases; route D trains the memo role alone from the same post-warm-up checkpoint, on the same inventories, at matched updates (Appendix C.12). Table 3 reports the final-round endpoints. We use the proposal pass rate (Pass) as a behavioral diagnostic of recognition.

Memo-only training buys coverage with faithfulness. D raises coverage at the cost of faithfulness and grounding (Figure 3a), at 4B even below the untrained base, although its reward penalizes rejected atoms. Raising the penalty only partly restores faithfulness (Appendix C.8). D's recognition of new atoms also stalls near the base: memo training only rewards atoms already in the inventory.

Table 3: APeB test set after round K ($K = 4$ at 4B, $K = 2$ at 35B-A3B). A alternates proposer and memo phases; θ_P is A before its final memo phase; D trains the memo role alone on the same inventories at matched updates; Base is the untrained checkpoint. Pass is the share of the proposer's held-out proposals that verification accepts; the other columns are memo metrics.

Model	Ckpt.	Pass	Cov.	Faith.	Ground.	Hit@1
4B	Base	30.7	30.7	74.5	74.9	57.8
	θ_P	**62.2**	64.6	**80.7**	**79.3**	61.8
	D	34.8	70.8	71.6	68.3	62.2
	A	61.1	**73.8**	77.1	74.7	**65.9**
35B-A3B	Base	42.4	47.7	75.7	77.8	62.5
	θ_P	**72.5**	70.4	**83.9**	**84.2**	65.1
	D	42.6	75.1	78.1	79.1	67.6
	A	68.9	**78.7**	81.7	81.9	**69.5**

Proposer training supplies what the memo reward cannot. Proposer GRPO raises faithfulness and grounding (Figure 3b), so θ_P has the highest of all checkpoints; its recognition of new atoms (Pass) also rises. Its coverage lags D, a task-switch gap of the memo readout (Appendix B.4). The memo phase then recovers coverage, keeps most of the recognition gain, and ends above D on every metric (Table 3, Figure 3d). Training recognition could relieve the recognition–expression bottleneck (Theorem 2).

PREMO scales to 35B, where faithfulness and grounding gain most. Qwen3.5-35B-A3B shows the same pattern (Figure 3c). Its base already recognizes and covers more atoms, so alternation adds less coverage than at 4B, consistent with Proposition 3. The trained 35B model improves both faithfulness and grounding over its base, whereas at 4B they barely improve, which supports attributing the remaining faithfulness gap of Table 1 to model scale.

4.4 RQ3: Does Multi-Round Discovery Add Useful Atoms (Figure 4)?

Figure 4 follows discovery round by round on the test sets: the new atoms the trained 4B proposer finds and GPT-5.4 accepts ($|N_k|$), the share of GPT-5.4's own four-round pool $E(x)$ that the inventory covers (Rec.), and memo coverage after each memo phase, for PREMO and for a control trained from the start on the final training-set inventory A_4^{train} (Fixed).

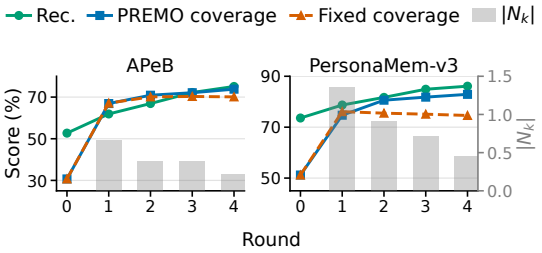


Figure 4: Per round on the APeB and PersonaMem-v3 test sets: frontier recall (Rec.), memo coverage of PREMO and of the Fixed control, and new atoms per instance $|N_k|$ (bars, right axis).

Rounds keep finding atoms, and memo training learns to express them. On noisy histories one round of discovery misses many atoms, and each later round finds new atoms that pass verification (Appendix C.7). After four rounds the trained 4B proposer has raised the inventory to three quarters of $E(x)$ on APeB and 86 percent on PersonaMem-v3. Memo coverage rises after every memo phase. The Fixed control, trained on the full inventory from the start, matches PREMO after one round and then saturates, while PREMO keeps rising. A full inventory is thus not enough: the model must first discover atoms as a proposer and then learn to express them. Discovery shrinks the undiscovered mass U_B (Proposition 1) and trains recognition first (RQ2).

4.5 RQ4: What Does Each Verification Step Contribute (Table 4, Figure 5)?

The verification path decides which proposals are accepted during proposer GRPO, and so what the proposer is rewarded for; it also decides which atoms enter the memo inventory. We run the full PREMO training with a single judge and with the path minus one step at a time (Table 4). Weaker verification lowers the faithfulness and grounding of the memo policy. Figure 5 follows memo training from the warm-up checkpoint on inventories built by a single judge, by the judge plus one step, and by the full path; each added step keeps faithfulness and grounding higher throughout training.

Table 4: APeB, 4B: PREMO trained at equal budget with each verifier variant. Path: the full path of Section 3.1 (ours); Single: one judge; w/o: the path without its support and counterexample checks (Verif.), merge and intent checks (Merge), or post-query weights (Weight).

			Path w/o		
	Path	Single	Verif.	Merge	Weight
Cov.	**73.8**	68.1	72.9	73.4	73.7
Faith.	**77.1**	71.2	72.3	74.1	77.0
Ground.	**74.7**	66.3	68.7	70.1	74.2
Hit@1	**65.9**	61.1	62.3	62.8	63.6

A single judge is not enough, and each step of the path pays. A single judge often trusts the cited records and misses rivals elsewhere in the history, so it accepts about 1.5 times as many proposals as the path, and its verdicts vary across calls (Appendix C.9). Trained with the single judge or with a check removed, PREMO writes memos that are worse on every metric, especially faithfulness and grounding. A simplified path during proposer GRPO rewards untrustworthy pro-

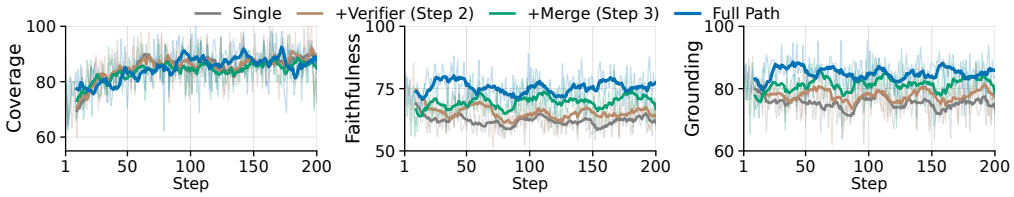


Figure 5: Qwen3.5-4B on APeB: memo GRPO from the post-warm-up checkpoint on inventories built by a single judge, by the judge plus the verifier (Step 2) or merge (Step 3), and by the full path with post-query weights; training-prompt curves (Appendix C.9). Coverage rises alike; each added step keeps faithfulness and grounding higher, the full path highest.

posals and weakens the recognition the proposer learns (Appendix C.9). Removing the post-query weights leaves the memo metrics unchanged and lowers Hit@1 by a smaller margin; the weights mainly change how firmly the memo states each preference (Appendix C.9). Figure 5 shows the same trend throughout memo training: coverage rises alike on every inventory, while faithfulness and grounding stay higher with each added step.

5 Related Work

Personalization from user histories. Language models are personalized with profiles, retrieved memories, behavioral histories, or explicit preference feedback (Salemi et al., 2024; Zhao et al., 2025; Yang et al., 2026; Rendle et al., 2009; Kang & McAuley, 2018), and recent methods maintain user memories, optimize natural-language preference representations against downstream reward (Chhikara et al., 2025; Xu et al., 2025b; Chen et al., 2025; Liu et al., 2026; Liang et al., 2026), or learn query-dependent checks from the history (Seo & Lee, 2026). A user-level persona is a prior that does not say which preferences are active for a given request, and a raw history shifts evidence selection onto every downstream call. PREMO instead trains a memo policy to extract the query-conditioned preference state, and supervises it with atoms verified against the history.

Verifiable and rubric-based supervision. Verifiable rewards and process supervision enable reliable post-training when success criteria are known (Cobbe et al., 2021; Shao et al., 2024; Lightman et al., 2024); rubrics extend this to open-ended generation (Gunjal et al., 2026), with personalized variants derived from user signals (Zhang et al., 2026; Qiu et al., 2026). Closest to us, EvoLM and EvoRubric share a single-policy design in which rubric generation and policy optimization co-evolve (Li et al., 2026; Guan et al., 2026). SERPO similarly co-evolves query-specific rubrics and policy parameters during test-time reinforcement learning (Wang et al., 2026). Rubric evolution changes what the policy is rewarded for; PREMO additionally asks whether learning to discover preferences improves the shared model that expresses them. Its criteria also differ: each atom is a hypothesis about an unobserved user state, so proposing it is not enough; it must be grounded in the history and verified before it becomes supervision (Appendix D).

6 Conclusion

We asked how to extract reliable query-relevant preferences from long, noisy histories into a memo that a downstream policy can act on. We show that the memo risk decomposes into coverage and faithfulness risks, which rest on recognition and expression of preferences. We supervise the memo with an inventory of verified preference atoms. Memo training alone improves recognition only through expression: it learns to cover the inventory while losing faithfulness. PREMO therefore alternates one model between a proposer role and a memo role: each proposer phase adds verified atoms to the inventory and hands the memo phase a checkpoint with better recognition and faithfulness. PREMO improves preference extraction, and its memos improve downstream decisions over the full history and match or beat the memos of its frontier teacher; the gains extend to 35B.

AI USE STATEMENT

In this work, we used generative AI tools to assist with developing, stress-testing, and checking parts of the theoretical framework, including proposing candidate formulations, assumptions, and proof steps, and checking derivations. All mathematical claims and proofs were independently reviewed, revised, and finalized by the authors. We also used generative AI tools for generating synthetic data, namely the queries of PersonaMovieLens and the filtering of APeB sessions (Appendix C.1); for implementing parts of the method and evaluation pipeline; and for assisting with experiment design and the analysis of results. Frontier LLMs are also components of the method and evaluation, as verifiers, judges, and the teacher for warm-up (Appendices C.1, C.2, C.10, and C.12). Additionally, we used generative AI tools for editing text for readability and identifying relevant literature. We have reviewed all AI-assisted work: all code was tested by the authors, and all experimental conclusions were verified by the authors against the reported results. We take responsibility for the final content of this work, including any text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses two existing benchmarks of user interaction histories, APeB and PersonaMem-v3, and PersonaMovieLens, which we build from the public MovieLens 32M ratings with synthesized queries (Appendix C.1). We collected no new data from human subjects and used all data under its released terms. Memos summarize a user's preferences, so a deployed memo policy processes personal data. Verification against the history is designed to lower the risk of unsupported claims about a user, but does not remove it, and memos should be stored and used under the same access controls as the history itself. We foresee no ethical concerns beyond those common to personalization systems, such as reinforcing existing preferences.

REPRODUCIBILITY STATEMENT

All theoretical results state their assumptions in the main text; complete proofs are given in Appendices A and B. Implementation details of PREMO are in Appendix C: task construction, data splits, and history preprocessing (Appendix C.1); the evaluation protocol with metric definitions, seeds, and judge–human agreement (Appendix C.2); condensed prompts (Appendix E); the verification path and the identifiers of the frontier models it calls (Appendix C.10); the proposer and memo rewards (Appendix C.11); training configuration, hyperparameters, and hardware (Appendix C.12). Code, all prompts, and the constructed PersonaMovieLens queries and data splits will be released upon publication.

REFERENCES

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in language models without supervision. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=ETKGuby0hcs.

Yizhuo Chen, Xin Liu, Ruijie Wang, Zheng Li, Pei Chen, Changlong Yu, Qingyu Yin, Priyanka Nigam, Meng Jiang, and Bing Yin. POPI: Personalizing LLMs via optimized natural language preference inference. *arXiv preprint arXiv:2510.17881*, 2025. doi: 10.48550/arXiv.2510.17881. URL https://arxiv.org/abs/2510.17881.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025. doi: 10.48550/arXiv.2504.19413. URL https://arxiv.org/abs/2504.19413.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. doi: 10.48550/arXiv.2110.14168. URL https://arxiv.org/abs/2110.14168.

Zhenlong Dai, Chang Yao, Wenkang Han, Ying Yuan, Zhipeng Gao, and Jingyuan Chen. MPCoder: Multi-user personalized code generator with explicit and implicit style representation

learning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3765–3780. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.207. URL https://aclanthology.org/2024.acl-long.207/.

Xin Guan, Xiaomeng Hu, Shen Huang, Zhenyi Wang, Bo Zhang, Zijian Li, Pengjun Xie, Bo Liu, and Jiuxin Cao. EvoRubric: Self-evolving rubric-driven RL for open-ended generation. *arXiv preprint arXiv:2605.29847*, 2026. doi: 10.48550/arXiv.2605.29847. URL https://arxiv.org/abs/2605.29847.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/cfd7bee7a651ee9af525098ef67a9e45-Abstract-Conference.html.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://www.nature.com/articles/s41586-025-09422-z.

F. Maxwell Harper and Joseph A. Konstan. The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19:1–19:19, 2015. doi: 10.1145/2827872. URL https://doi.org/10.1145/2827872.

Yulin Hu, Zimo Long, Jiahe Guo, Xingyu Sui, Xing Fu, Weixiang Zhao, Yanyan Zhao, and Bing Qin. OP-Bench: Benchmarking over-personalization for memory-augmented personalized conversational agents. *arXiv preprint arXiv:2601.13722*, 2026. doi: 10.48550/arXiv.2601.13722. URL https://arxiv.org/abs/2601.13722.

IMDb. IMDb non-commercial datasets. https://developer.imdb.com/non-commercial-datasets/, 2026. Accessed 2026-09-14.

Bowen Jiang, Yuan Yuan, Zhuoqun Hao, Yuchen Liu, Maohao Shen, Sihao Chen, Gregory Wornell, Chris Callison-Burch, Lyle Ungar, Dan Roth, Qi Guo, Xiangjun Fan, Camillo J. Taylor, and Hanchao Yu. PersonaMem-v3: Toward omni-platform personal intelligence for holistic user understanding, recommendation, and agentic tasks. *arXiv preprint arXiv:2608.21381*, 2026. doi: 10.48550/arXiv.2608.21381. URL https://arxiv.org/abs/2608.21381.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206. IEEE, 2018. doi: 10.1109/ICDM.2018.00035. URL https://doi.org/10.1109/ICDM.2018.00035.

Anjali Kantharuban, Jeremiah Milbauer, Maarten Sap, Emma Strubell, and Graham Neubig. Stereotype or personalization? User identity biases chatbot recommendations. *arXiv preprint arXiv:2410.05613*, 2024. doi: 10.48550/arXiv.2410.05613. URL https://arxiv.org/abs/2410.05613.

Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 447–456. ACM, 2009. doi: 10.1145/1557019.1557072. URL https://doi.org/10.1145/1557019.1557072.

Shuyue Stella Li, Rui Xin, Teng Xiao, Yike Wang, Rulin Shao, Zoey Hao, Melanie Sclar, Sewoong Oh, Faeze Brahman, Pang Wei Koh, and Yulia Tsvetkov. EvoLM: Self-evolving language models through co-evolved discriminative rubrics. *arXiv preprint arXiv:2605.03871*, 2026. doi: 10.48550/arXiv.2605.03871. URL https://arxiv.org/abs/2605.03871.

Kaiqu Liang, Julia Kruk, Shengyi Qian, Xianjun Yang, Shengjie Bi, Yuanshun Yao, Shaoliang Nie, Mingyang Zhang, Lijuan Liu, Jaime Fernández Fisac, Shuyan Zhou, and Saghar Hosseini. Learning personalized agents from human feedback. *arXiv preprint arXiv:2602.16173*, 2026. doi: 10.48550/arXiv.2602.16173. URL https://arxiv.org/abs/2602.16173.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Yuting Liu, Jian Guan, Jia-Nan Li, Wei Wu, Jiang-Ming Yang, Jianzhe Zhao, and Guibing Guo. Text as a universal interface for transferable personalization. *arXiv preprint arXiv:2601.04963*, 2026. doi: 10.48550/arXiv.2601.04963. URL https://arxiv.org/abs/2601.04963.

Yilun Qiu, Xiaoyan Zhao, Yang Zhang, Yuxin Chen, Cilin Yan, Jiayin Cai, Xiaolong Jiang, Yao Hu, Yoko Yamakata, and Tat-Seng Chua. Preference-aware rubric learning for personalized evaluation. *arXiv preprint arXiv:2605.31545*, 2026. doi: 10.48550/arXiv.2605.31545. URL https://arxiv.org/abs/2605.31545.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 452–461. AUAI Press, 2009.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. LaMP: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7370–7392. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.399. URL https://aclanthology.org/2024.acl-long.399/.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2014. doi: 10.48550/arXiv.1312.6120. URL https://arxiv.org/abs/1312.6120v3.

Kwangwook Seo and Dongha Lee. P-Check: Advancing personalized reward model via learning to generate dynamic checklist. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43447–43471, 2026. URL https://aclanthology.org/2026.acl-long.2011/.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. doi: 10.48550/arXiv.2402.03300. URL https://arxiv.org/abs/2402.03300.

Yushi Sun, Yanjie Zhang, and Rui Sheng. The personalization mirage: How LLMs fabricate user profiles, and why self-monitoring misleads. *arXiv preprint arXiv:2608.04570*, 2026. URL https://arxiv.org/abs/2608.04570.

Jianze Wang, Kunwang Zheng, Ying Liu, Yu Cao, Qilong Zhang, Jinlong Chen, Hua Yang, and Qianglong Chen. SERPO: Self-evolving rubric policy optimization for open-ended test-time reinforcement learning. *arXiv preprint arXiv:2607.26873*, 2026. doi: 10.48550/arXiv.2607.26873. URL https://arxiv.org/abs/2607.26873.

Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. Denoising implicit feedback for recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 373–381. ACM, 2021. doi: 10.1145/3437963.3441800. URL https://doi.org/10.1145/3437963.3441800.

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Hu, Jie Huang, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, and Quoc V. Le. Long-form factuality in large language models. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2403.18802.

Austin Xu, Srijan Bansal, Yifei Ming, Semih Yavuz, and Shafiq Joty. Does context matter? ContextualJudgeBench for evaluating LLM-based judges in contextual settings. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9541–9564, 2025a. doi: 10.18653/v1/2025.acl-long.470. URL https://aclanthology.org/2025.acl-long.470/.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, 2025b. URL https://arxiv.org/abs/2502.12110.

Garry Yang, Zizhe Chen, Xinru Chen, Yongqiang Chen, Jianxiang Wang, Deyu Zou, Linyi Ding, Jialiang Wu, Yunzhong He, Yu Gong, James Cheng, and Huaixiao Tou. APeB: Benchmarking personalization ability of large language model agents. *arXiv preprint arXiv:2607.03162*, 2026. doi: 10.48550/arXiv.2607.03162. URL https://arxiv.org/abs/2607.03162.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025. URL https://arxiv.org/abs/2503.14476.

Pinyi Zhang, Ting-En Lin, Yuchuan Wu, Jingyang Chen, Zongqi Wang, Hua Yang, Bing Zhao, Fei Huang, Yongbin Li, and Kai Zhang. P-GenRM: Personalized generative reward model with test-time user-based scaling. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/73bda9a20f6f9f6074ce822e76f126bb-Abstract-Conference.html.

Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. Do LLMs recognize your preferences? evaluating personalized preference following in LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/28a46044775d97a4efcbcf14e7f13209-Abstract-Conference.html.

A MEMO RISK DECOMPOSITION, BOTTLENECK DYNAMICS, AND ACTUAL-UPDATE TRANSFER

This appendix proves the results of Section 2: Proposition 1 and its per-round refinement used in RQ2 and RQ3 (Appendix A.1), Theorem 2 and Proposition 3 (Appendix A.2), a trainable proposer readout (Appendix A.3), the transfer under actual GRPO updates, Theorem A.4 (Appendix A.4), and the bridge from the effective gain g to the risks of Eq. 1 (Appendix A.5). Throughout, the atom sets, the weights $a_{x,r}$, and the verifier are fixed; the route comparison also fixes supervision access, initialization, and the update budget.

A.1 MEMO RISK DECOMPOSITION AND ITS PER-ROUND REFINEMENT

Write $\mu(B) = \mathbb{E}_x \sum_{r \in B(x)} a_{x,r}$ and assume finite expected total weight. All claims are evaluated with a fixed atomization, semantic matching rule, and history verifier. Validity requires support and query relevance. The recognition probability $p_\theta(r \mid x)$ of Section 2 counts assertions that r is valid under a fixed prompt using only $x = (Q, H)$; whether r is proposed at all is a separate event (Appendix B.2). Define

$$\begin{aligned} C_P^B &= \mathbb{E}_x \sum_{r \in B(x)} a_{x,r}(1 - p_\theta), \quad & D_B^+ &= \mathbb{E}_x \sum_{r \in B(x)} a_{x,r}[p_\theta - m_\theta]_+, \\ H_P &= \mathbb{E}_x \sum_{r \in S^-(x)} a_{x,r} p_\theta, & D^- &= \mathbb{E}_x \sum_{r \in S^-(x)} a_{x,r}[m_\theta - p_\theta]_+. \end{aligned} \tag{7}$$

The probabilities in each summand are evaluated at $(r \mid x)$.

Lemma A.1 (Undiscovered mass)**.** *For any valid subset $A \subseteq \mathcal{F}^+$ and $U = \mu(\mathcal{F}^+ \setminus A)$,*

$$C_M^A \le C_M^{\mathcal{F}^+} \le C_M^A + U.$$

Proof. Split the valid set into A and its complement and use $0 \le 1 - m_\theta \le 1$ on the latter. □

Lemma A.2 (Negative faithfulness and positive coadaptation)**.** *For a valid subset $B \subseteq \mathcal{F}^+$, the envelopes satisfy*

$$\begin{aligned} C_M^B \le \Psi_B^+ &= \mathbb{E}_x \sum_{r \in B(x)} a_{x,r}[1 - \min(p_\theta, m_\theta)] = C_P^B + D_B^+ \le \mu(B), \\ H_M \le \Psi^- &= \mathbb{E}_x \sum_{r \in S^-(x)} a_{x,r} \max(p_\theta, m_\theta) = H_P + D^- \le \mu(S^-). \end{aligned} \tag{8}$$

For a fixed valid subset $N \subseteq \mathcal{F}^+$ and two checkpoints θ_P, θ_{PM} before and after a memo-training phase, define $G_{\mathrm{coad}} = \Psi_N^+(\theta_P) - \Psi_N^+(\theta_{PM})$. Then $C_M^N(\theta_{PM}) \le \Psi_N^+(\theta_P) - G_{\mathrm{coad}}$.

Proof. The identities follow pointwise from $1 - \min(p, m) = (1 - p) + [p - m]_+$ and $\max(p, m) = p + [m - p]_+$. The inequalities follow from $1 - m \le 1 - \min(p, m)$ and $m \le \max(p, m)$. Substituting the definition of G_{coad} proves the final statement. □

Proof of Proposition 1. For any $B \subseteq \mathcal{F}^+$, let $U_B = \mu(\mathcal{F}^+ \setminus B)$. Lemma A.1 and Lemma A.2 give

$$R_M(\theta) = C_M^B(\theta) + C_M^{\mathcal{F}^+ \setminus B}(\theta) + \lambda H_M(\theta) \le U_B + \Psi_B^+(\theta) + \lambda \Psi^-(\theta).$$

□

Round-specific refinement. For the training rounds of Section 3.4, let A_k contain previously accepted atoms and N_k contain newly accepted atoms not in A_k. Assume these sets are disjoint subsets of $\mathcal{F}^+$, i.e., verification admits no invalid atom. Write $\mathcal{U}_{k+1} = \mathcal{F}^+ \setminus (A_k \cup N_k)$, $U_{k+1} = \mu(\mathcal{U}_{k+1})$ (the U_B of RQ3 with $B = A_{k+1}$), and $U_k = \mu(\mathcal{F}^+ \setminus A_k)$. Keeping the old-atom coverage risk exact yields the sharper bound

$$R_M(\theta) \le C_M^{A_k}(\theta) + U_{k+1} + \Psi_{N_k}^+(\theta) + \lambda \Psi^-(\theta). \tag{9}$$

Indeed, split $\mathcal{F}^+$ into A_k, N_k, and $\mathcal{U}_{k+1}$, apply the coverage envelope on N_k, and charge the remaining weight in full. Discovering an atom replaces its worst-case omission cost with a model-dependent envelope; a small envelope requires both recognition and expression to be accurate. Here θ_P and θ_{PM} are the checkpoints after the round's proposer and memo phases (Eq. 6); G_{coad} is evaluated on the fixed set N_k.

The certificate also admits an exact time decomposition. Put $\overline{C}_k(\theta_P) = C_M^{A_k}(\theta_P) + U_k$ and $\overline{C}_{k+1}(\theta_{PM}) = C_M^{A_k}(\theta_{PM}) + U_{k+1} + \Psi^+_{N_k}(\theta_{PM})$. Since $U_k - U_{k+1} = \mu(N_k)$,

$$\begin{aligned}\overline{C}_k(\theta_P) - \overline{C}_{k+1}(\theta_{PM}) = C_M^{A_k}(\theta_P) - C_M^{A_k}(\theta_{PM}) \\ + \mu(N_k) - \Psi^+_{N_k}(\theta_P) + G_{\mathrm{coad}}.\end{aligned} \tag{10}$$

Thus old-preference damage and unsuccessful recovery are represented explicitly. Estimating U_{k+1} needs an audit of the valid set; RQ3's frontier recall, the share of the GPT-5.4 pool that the inventory covers, serves this role.

A.2 Fixed-Readout Recognition–Expression Bottleneck

Multiplicative representation/readout dynamics are a tractable model of learning delays (Saxe et al., 2014). We fix coordinates, readout scales, loss scales, and optimization time before comparing routes; for positive scalar factors let $g = uv$ be the memo gain. The squared-error objectives and negative gradients are

$$\begin{aligned} L_M(u,v) &= \tfrac{1}{2}(1-uv)^2, \quad f_M(u,v) = (v(1-uv), u(1-uv)), \\ L_P(u) &= \tfrac{1}{2}(1-u)^2, \qquad f_P(u,v) = (1-u, 0). \end{aligned} \tag{11}$$

These objectives define the effective dynamics; actual GRPO updates are handled in Appendix A.4. Rescaling $u \mapsto au, v \mapsto v/a$ preserves g but changes Euclidean gradient flow, so the scale convention is part of the model.

Existence and invariant. The vector fields are locally Lipschitz. Along memo flow,

$$\frac{d}{dt}(u^2 - v^2) = 0, \qquad \dot{g} = s(1-g), \qquad s = u^2 + v^2 = \sqrt{(u^2-v^2)^2 + 4g^2}. \tag{12}$$

On every local interval with $g_0 < 1$,

$$1 - g(t) = (1-g_0)\exp\left[-\int_0^t s(a)\,da\right] > 0.$$

Thus g increases and remains below one. The invariant and bounded g bound s, excluding finite-time explosion and giving a solution at every finite time. Positive initial u, v stay positive.

Proof of Theorem 2. Under memo-only training, $d = u - v$ satisfies $\dot{d} = -(1-uv)d$. The initial equality $u_0 = v_0 = \sigma$ is preserved, so $\dot{g}_D = 2g_D(1-g_D)$ and $g_D(0) = \sigma^2$. Integrating $\frac{d}{dt}\log[g_D/(1-g_D)] = 2$ gives

$$g_D(t) = \frac{1}{1 + (\sigma^{-2} - 1)e^{-2t}},$$

which is Eq. 2. In particular, its initial increase is $\dot{g}_D(0) = 2\sigma^2(1-\sigma^2)$. For any fixed target $g_\star \in (0,1)$ and $0 < \sigma < \sqrt{g_\star}$, solving $g_D(\tau_\star) = g_\star$ gives

$$\tau_\star = \frac{1}{2}\log\left[\frac{g_\star(1-\sigma^2)}{\sigma^2(1-g_\star)}\right] = \log(1/\sigma) + O(1) \quad \text{as } \sigma \to 0^+. \tag{13}$$

Thus reaching any fixed positive gain requires a time that diverges as the initial factors become weak. □

Proof of Proposition 3. Let route A use auxiliary recognition training for $t_P > 0$ and then memo training for $t_M > 0$; route D uses only memo training for $T = t_P + t_M$. Both start from $(u, v) =$

(σ, σ). Auxiliary recognition flow has the solution $u(t) = 1 - (1-\sigma)e^{-t}$ and $v(t) = \sigma$. Put $\alpha = 1 - (1-\sigma)e^{-t_P}$ and $c = \alpha^2 - \sigma^2$. At the switch, $\sigma < \alpha < 1$ and $g_0 = \alpha\sigma \in (0,1)$. During subsequent memo training, $u^2 - v^2 = c > 0$, so $s = c + 2v^2 \geq c$. Since $1 - g > 0$,

$$\frac{d}{dt}(1-g) \leq -c(1-g), \qquad 1 - g_A \leq (1-\alpha\sigma)e^{-ct_M}.$$

This proves the lower bound $\underline{g}_A = 1 - (1-\alpha\sigma)e^{-ct_M}$ in Eq. 3. Define $\Delta_{\mathrm{bn}} = \underline{g}_A - g_D(T)$. A positive Δ_{bn} gives $g_A > g_D(T)$, and $L_M = \frac{1}{2}(1-g)^2$ is strictly decreasing for $g < 1$. In particular, $L_M(D) - L_M(A) = \frac{1}{2}(g_A - g_D)(2 - g_A - g_D) > 0$.

For nonemptiness, fix $t_P, t_M > 0$ and set $a_* = 1 - e^{-t_P} > 0$. The direct gain tends to zero as $\sigma \to 0^+$, whereas $\underline{g}_A \to 1 - e^{-a_*^2 t_M} > 0$. Their difference extends continuously to $\sigma = 0$ with a positive value. It is therefore positive throughout some interval $(0, \sigma_0)$. □

The mechanism concerns finite-time learning: both routes approach $g = 1$ with long enough memo training.

A.3 Trainable Proposer Readout and Robustness

Let the proposer readout w also train, with

$$L_P(u,w) = \tfrac{1}{2}(1-wu)^2, \qquad \dot{u} = w(1-wu), \quad \dot{w} = u(1-wu), \quad \dot{v} = 0. \tag{14}$$

Both routes start from $(u, v, w) = (\sigma, \sigma, w_0)$, where $w_0 > 0$ is fixed and $0 < \sigma < \min(1, w_0, 1/w_0)$. Route D leaves w fixed.

Theorem A.3 (Bottleneck with a trainable proposer readout)**.** *Set $c_P = w_0^2 - \sigma^2$ and define*

$$\underline{\zeta}_P = 1 - (1 - w_0\sigma)e^{-c_P t_P}, \qquad \underline{u}_P^2 = \frac{2\underline{\zeta}_P^2}{\sqrt{c_P^2 + 4\underline{\zeta}_P^2} + c_P}. \tag{15}$$

With $B(a) = 1 - (1 - a\sigma)e^{-(a^2-\sigma^2)t_M}$, one has $g_A \geq B(\underline{u}_P)$. If $B(\underline{u}_P) > g_D(t_P + t_M)$, the alternating route has larger gain and lower memo loss. For every fixed $w_0, t_P, t_M > 0$, the condition holds throughout a sufficiently small positive σ interval.

Proof. Proposer flow preserves $w^2 - u^2 = c_P > 0$. Its product $\zeta = wu$ satisfies $\dot{\zeta} = (w^2 + u^2)(1 - \zeta)$, with $0 < \zeta_0 = w_0\sigma < 1$. The same integral and boundedness argument as for memo flow gives a global finite-time solution with positive $u, w, \zeta < 1$, and $u < 1$. Since $w^2 + u^2 \geq c_P$, integration yields $\zeta_P \geq \underline{\zeta}_P$.

The map $u \mapsto u\sqrt{u^2 + c_P}$ is strictly increasing on $u \geq 0$. Solving $\zeta^2 = u^2(u^2 + c_P)$ and rationalizing gives Eq. 15. Hence $\sigma < \underline{u}_P \leq u_P < 1$. Recovery from (u_P, σ) obeys $g_A \geq B(u_P)$. For $\sigma \leq a < 1/\sigma$,

$$B'(a) = e^{-(a^2-\sigma^2)t_M}[\sigma + 2at_M(1 - a\sigma)] > 0,$$

so $g_A \geq B(\underline{u}_P)$.

As $\sigma \to 0^+$, $c_P \to w_0^2$ and $\underline{\zeta}_P \to \zeta_* = 1 - e^{-w_0^2 t_P} > 0$. Therefore

$$\underline{u}_P^2 \longrightarrow u_*^2 = \frac{2\zeta_*^2}{\sqrt{w_0^4 + 4\zeta_*^2} + w_0^2} > 0.$$

The alternating lower bound tends to $1 - e^{-u_*^2 t_M} > 0$, while direct gain tends to zero at every fixed finite time. Continuity gives the claimed interval within the stated domain. □

Unequal costs. For fixed positive unit costs κ_P, κ_M, match $\kappa_P t_P + \kappa_M t_M = \kappa_M T_D$, giving $T_D = t_M + (\kappa_P/\kappa_M)t_P$. Both nonemptiness proofs still hold for every fixed finite cost ratio. The experiments match parameter updates (Appendix C.12).

Perturbed recovery and initialization. Suppose a separately specified recovery state is (α, b), with $\alpha > 0$, $|b| < \alpha$, and $-1 < \alpha b < 1$. The invariant $u^2 - v^2 = \alpha^2 - b^2 > 0$ keeps $u > 0$. The gain increases, remains below one, and satisfies

$$g(t_M) \geq 1 - (1 - \alpha b)e^{-(\alpha^2 - b^2)t_M}.$$

This includes negative readouts and follows from the same positive $1-g$ comparison. By continuous dependence on initial conditions, a strict margin persists in a neighborhood of any favorable example. A weak proposer readout, substantial role interference (the task-switch gap of Appendix B.4), or insufficient recovery can eliminate the advantage.

A.4 Paired Updates and the Route Replacement Identity

Fix the valid and invalid atom sets, their weights, and supervision access throughout the comparison. The final risks are $R_C = C_M^{\mathcal{F}^+}$ and $R_H = H_M$. Route A applies proposer updates followed by memo updates, whereas route D applies only memo updates, as in Section 3.4; both start from the same complete training state and take the same number of parameter updates. Both roles use response-level GRPO, including its normalization, clipping, regularization, and optimizer effects.

The complete state Z includes every variable needed for a Markov training transition, including optimizer state, time, and any data-dependent selection state. Fix the units of training and their budgets in advance. Updates use fixed state coordinates and norm.

Quantities in Theorem A.4. Routes A and D take n update steps from a common state Z_0. At step ℓ with state Z_ℓ on route A, let $\Delta_{A,\ell}$ be the update A takes, $\Delta_{D,\ell}$ the memo update D would take from Z_ℓ, $d_\ell = \Delta_{A,\ell} - \Delta_{D,\ell}$, and $F^D_{\ell+1,j}(Z)$, $j \in \{C, H\}$, the expected final risk R_j if direct training completes the remaining steps from Z, with $L_{\ell,j}$-Lipschitz gradient. With $\mathbb{E}_A$ the expectation along route A, $G_{\mathrm{tr},j} = -\sum_\ell \mathbb{E}_A\langle \nabla F^D_{\ell+1,j}(Z_\ell), d_\ell\rangle$ is the accumulated first-order risk reduction of A's updates over D's, and $B_{\mathrm{tr},j} = \frac{1}{2}\sum_\ell \mathbb{E}_A[L_{\ell,j}(\|\Delta_{A,\ell}\| + \|\Delta_{D,\ell}\|)\|d_\ell\|]$ is the curvature remainder.

Theorem A.4 (Transfer under actual updates)**.** *Assume the integrability, coupling, and smoothness conditions of Lemmas A.5 and A.6. Also assume the aligned readout, projected-dynamics approximation, and risk-residual bounds of Appendix A.5, using ideal routes with the initialization and durations of Proposition 3. Let W_j be the modeled subset weights defined there and let $\epsilon_{\mathrm{eff},j}$ be the total approximation error in Eq. 22. Then, for $j \in \{C, H\}$,*

$$\mathbb{E}R_j(D) - \mathbb{E}R_j(A) \geq \max\left\{G_{\mathrm{tr},j} - B_{\mathrm{tr},j},\ \frac{W_j \Delta_{\mathrm{bn}}}{2} - \epsilon_{\mathrm{eff},j}\right\}. \tag{16}$$

The first bound uses the paired-update conditions, while the second uses the approximation and readout conditions. Either gives strict improvement when its lower bound is positive; the two bounds are alternatives, and their gains are not added. Both risks improve when this holds separately for $j = C$ and $j = H$.

Lemma A.5 (Paired finite-continuation comparison)**.** *At a common state Z, let Δ_A, Δ_D be actual equal-budget updates under a coupling preserving their marginal laws, and let F be the expected risk after the same prescribed finite continuation. Suppose F has an L-Lipschitz gradient on a convex region containing Z, $Z + \Delta_A$, and $Z + \Delta_D$. With finite second moments and integrable remainder, define*

$$G_F = -\langle \nabla F(Z), \mathbb{E}[\Delta_A - \Delta_D \mid Z]\rangle, \quad B_F = \frac{1}{2}\mathbb{E}[L(\|\Delta_A\| + \|\Delta_D\|)\|\Delta_A - \Delta_D\| \mid Z].$$

Then

$$-G_F - B_F \leq \mathbb{E}[F(Z + \Delta_A) - F(Z + \Delta_D) \mid Z] \leq -G_F + B_F.$$

Proof. Put $d = \Delta_A - \Delta_D$. The fundamental theorem of calculus gives

$$F(Z + \Delta_A) - F(Z + \Delta_D) = \langle \nabla F(Z), d\rangle + e,$$

where

$$|e| \leq L\|d\| \int_0^1 \|(1 - \tau)\Delta_D + \tau\Delta_A\|\, d\tau \leq \frac{L}{2}(\|\Delta_A\| + \|\Delta_D\|)\|d\|.$$

Take conditional expectations. Actual normalization, clipping, KL terms, Adam state, and sampling remain in the updates. No unbiased-gradient assumption is used. □

Lemma A.6 (Route replacement)**.** *For each step $\ell = 0, \dots, n-1$, let $K_{A,\ell}$ and $K_{D,\ell}$ be the actual transition kernels. Assume a common initial state and an integrable terminal risk R_j. Define $F^D_{n,j} = R_j$ and $F^D_{\ell,j} = K_{D,\ell} F^D_{\ell+1,j}$. At states Z^A_ℓ along the A prefix, let $D_{\ell,j} = (K_{A,\ell} - K_{D,\ell}) F^D_{\ell+1,j}$. Then*

$$\mathbb{E}R_j(A) - \mathbb{E}R_j(D) = \sum_{\ell=0}^{n-1} \mathbb{E}_A D_{\ell,j}(Z^A_\ell). \tag{17}$$

Proof. Conditional expectation yields

$$\mathbb{E}_A[F^D_{\ell+1,j}(Z^A_{\ell+1}) - F^D_{\ell,j}(Z^A_\ell)] = \mathbb{E}_A D_{\ell,j}(Z^A_\ell).$$

Summing telescopes to $\mathbb{E}_A F^D_{n,j}(Z^A_n) - F^D_{0,j}(Z_0)$, which is the difference of terminal risks. □

Proof of Theorem A.4, first term in Eq. 16. Apply Lemma A.5 to $F^D_{\ell+1,j}$ and the paired actual updates at every A-prefix state. Average and sum, using Lemma A.6, to obtain

$$-G_{\mathrm{tr},j} - B_{\mathrm{tr},j} \le \mathbb{E}R_j(A) - \mathbb{E}R_j(D) \le -G_{\mathrm{tr},j} + B_{\mathrm{tr},j}.$$

The upper bound, negated, is the first term of Eq. 16; it is strictly positive whenever $G_{\mathrm{tr},j} > B_{\mathrm{tr},j}$. Apply this separately to $j = C, H$. Only the cumulative margin must be positive. □

Because F already evaluates the final memo after its finite continuation, its risk includes retention and expression effects. The identities in Appendix A.1 provide complementary diagnostics. Table 3 reports the left-hand side of Eq. 16 at matched updates.

A.5 From the Effective Model to Actual Memo Risks

Aligned risk readout. Fix feasible subsets $B^+ \subseteq \mathcal{F}^+$ and $B^- \subseteq S^-$ whose claims can jointly respect memo length and consistency constraints. Let $W_C = \mu(B^+) > 0$, $W_H = \mu(B^-) > 0$. For $-1 \le g \le 1$, assume an aligned probabilistic readout $\widetilde{m}_g = (1+g)/2$ on B^+ and $\widetilde{m}_g = (1-g)/2$ on B^-. It gives

$$\widetilde{R}_j(g) = \frac{W_j(1-g)}{2}, \qquad \widetilde{R}_j(g_A) - \widetilde{R}_j(g_D) \le -\frac{W_j \Delta_{\mathrm{bn}}}{2}, \quad j \in \{C, H\}. \tag{18}$$

The claim follows by summing valid omissions and invalid inclusions.

Projected actual dynamics. Choose a fixed projection $z = (u, v) = \psi(Z)$ before observing the endpoints, and let $G(z) = uv$ map the projected state to its gain. The ideal trajectories start at (σ, σ) and use the auxiliary recognition and memo fields of Eq. 11, with the durations of Proposition 3. On a common bounded convex region Ω, assume each effective role field f_r is L_f-Lipschitz and $\|f_r\| \le \bar{f}$. Actual projected iterates, ideal trajectories, and required line segments stay in this region. Align the time grid with all role switches. For each route suppose

$$z_{i+1} = z_i + h f_{r_i}(z_i) + h e_i, \qquad \mathbb{E}\|e_i\| \le \nu_i, \qquad \mathbb{E}\|z_0 - z_0^*\| \le \Xi_0. \tag{19}$$

The residual includes bias and stochasticity. This projection accommodates a full-state optimizer through Z.

Discrete error bound. The ideal flow satisfies $z^*(t_{i+1}) = z^*(t_i) + h f_{r_i}(z^*(t_i)) + \rho_i$. Its integral equation gives $\|\rho_i\| \le \int_0^h L_f \bar{f} s \, ds = L_f \bar{f} h^2/2$. Writing $\Xi_i = \mathbb{E}\|z_i - z^*(t_i)\|$ therefore yields

$$\begin{aligned} \Xi_{i+1} &\le (1 + hL_f)\Xi_i + h\nu_i + \tfrac{1}{2} L_f \bar{f} h^2, \\ \Xi_N &\le a^N \Xi_0 + \sum_{i=0}^{N-1} a^{N-1-i}(h\nu_i + \tfrac{1}{2} L_f \bar{f} h^2), \qquad a = 1 + hL_f. \end{aligned} \tag{20}$$

For each route, Nh is its declared effective duration.

Risk-mapping and unmodeled content. Write the true fixed-set risk as $R_j^B + R_j^{\mathrm{rest}}$. Define the risk on projected states by $\overline{R}_j(z) = \widetilde{R}_j(G(z))$ on its readout domain. Assume $\overline{R}_j$ has a K_j-Lipschitz extension to Ω and that for $s \in \{A, D\}$,

$$\begin{aligned} &|\mathbb{E}R_j^B(Z_s) - \mathbb{E}\overline{R}_j(\psi(Z_s))| \le \chi_{s,j}, \\ &\mathbb{E}R_j^{\mathrm{rest}}(Z_A) - \mathbb{E}R_j^{\mathrm{rest}}(Z_D) \le \omega_j, \qquad \omega_j \ge 0. \end{aligned} \tag{21}$$

Let Ξ_A, Ξ_D be the bounds from Eq. 20. Adding and subtracting the projected and ideal endpoint risks gives

$$\begin{aligned} \mathbb{E}R_j(A) - \mathbb{E}R_j(D) &\le -\frac{W_j \Delta_{\mathrm{bn}}}{2} + \epsilon_{\mathrm{eff},j}, \\ \epsilon_{\mathrm{eff},j} &= K_j(\Xi_A + \Xi_D) + \chi_{A,j} + \chi_{D,j} + \omega_j. \end{aligned} \tag{22}$$

Indeed each ideal-versus-projected risk differs by at most $K_j \Xi_s$; Eq. 21 controls the remaining endpoint differences, and Eq. 18 supplies the ideal gain. Negated, this is the second term in Eq. 16 of Theorem A.4.

B The Memo-versus-History Bound and the Task-Switch Gap

This appendix derives the memo-versus-history bound used in Section 2 and RQ1 (Eq. 55): a query-conditioned preference state is sufficient for the decision (Appendix B.1), verification recovers it atom by atom (Appendix B.2), and a memo expressing the recovered atoms loses at most a bounded oracle risk (Appendix B.3). Appendix B.4 explains the temporary memo degradation at θ_P; the route comparison under actual updates is Appendix A.4. Throughout, risks and objectives are minimized (a maximized reward enters as its negative expectation), and expectations run over the data and over sampling and optimization randomness. For Proposition B.1 we assume the standard measurability and attainment conditions for a Bayes action.

B.1 State-Mediated Task Factorization

For user u, let $H_t = \mathcal{H}_{u,\le t}$ be the observed trajectory at time t, and let Q_t, C_t, and Y_t denote the current query, candidate actions, and desired decision; we suppress u below. Events in H_t are noisy observations of a latent, time-varying user state Z_t: they may reflect exposure, price, inventory, temporary intent, or decisions for others.

The *active preference state* is the part of the current state relevant to a particular query, obtained by a query-dependent projection Π:

$$S_t^\star(q) = \Pi(Z_t, q). \tag{23}$$

Thus, the same history can induce different active states for different queries, and the state for a fixed query can change as the trajectory evolves. We assume that the ideal active state is sufficient for the current decision:

$$Y_t \perp\!\!\!\perp H_t \mid Q_t, C_t, S_t^\star(Q_t). \tag{24}$$

Proposition B.1 (State-mediated task factorization)**.** *Under Eq. 24, for any decision loss there exists a Bayes-optimal policy using only $(Q_t, C_t, S_t^\star(Q_t))$; residual details in H_t cannot improve its Bayes risk.*

Proof. For compactness, define

$$X_t = \big(Q_t,\, C_t,\, S_t^\star(Q_t)\big).$$

Let $\mathcal{D}$ be the decision space and let

$$\ell : \mathcal{D} \times \mathcal{Y} \to \mathbb{R}$$

be any integrable decision loss. Equation 24 states that

$$Y_t \perp\!\!\!\perp H_t \mid X_t.$$

Equivalently, the conditional law of Y_t satisfies

$$\mathbb{P}(Y_t \in \mathcal{B} \mid X_t, H_t) = \mathbb{P}(Y_t \in \mathcal{B} \mid X_t) \tag{25}$$

almost surely for every measurable set $\mathcal{B} \subseteq \mathcal{Y}$.

For a fixed value x of X_t and a decision $d \in \mathcal{D}$, define the conditional risk

$$\rho(d, x) = \mathbb{E}[\ell(d, Y_t) \mid X_t = x]$$

and its Bayes envelope

$$\rho^\star(x) = \inf_{d \in \mathcal{D}} \rho(d, x).$$

Consider an arbitrary policy

$$\delta_{\text{full}} : (X_t, H_t) \mapsto \mathcal{D}$$

that is allowed to observe the full trajectory in addition to the active state. By Eq. 25,

$$\begin{aligned}
&\mathbb{E}[\ell(\delta_{\text{full}}(X_t, H_t), Y_t) \,|\, X_t, H_t] \\
&\quad = \int_{\mathcal{Y}} \ell(\delta_{\text{full}}(X_t, H_t), y)\, \mathbb{P}(dy \mid X_t) \\
&\quad = \rho(\delta_{\text{full}}(X_t, H_t), X_t) \\
&\quad \geq \rho^\star(X_t).
\end{aligned} \tag{26}$$

Taking expectations gives

$$\mathbb{E}[\ell(\delta_{\text{full}}(X_t, H_t), Y_t)] \geq \mathbb{E}[\rho^\star(X_t)]\,. \tag{27}$$

Under the stated attainment and measurable-selection conditions, choose a measurable Bayes action

$$\delta^\star(x) \in \arg\min_{d \in \mathcal{D}} \rho(d, x).$$

This policy depends only on X_t, hence only on

$$\big(Q_t, C_t, S_t^\star(Q_t)\big).$$

Its risk is

$$\begin{aligned}
\mathbb{E}[\ell(\delta^\star(X_t), Y_t)] &= \mathbb{E}[\rho(\delta^\star(X_t), X_t)] \\
&= \mathbb{E}[\rho^\star(X_t)]\,.
\end{aligned} \tag{28}$$

Combining Eqs. 27 and 28 yields

$$\begin{aligned}
&\inf_{\delta_{\text{full}}} \mathbb{E}[\ell(\delta_{\text{full}}(X_t, H_t), Y_t)] \\
&\qquad = \inf_{\delta_{\text{state}}} \mathbb{E}[\ell(\delta_{\text{state}}(X_t), Y_t)] \\
&\qquad = \mathbb{E}[\rho^\star(X_t)]\,.
\end{aligned} \tag{29}$$

Thus, once the ideal active preference state is observed, direct access to the remaining details of H_t cannot reduce Bayes risk. A Bayes-optimal policy can therefore be expressed using only $(Q_t, C_t, S_t^\star(Q_t))$. □

B.2 Verifier-Backed Atomic Recovery

Because $S_t^\star(q)$ is latent, we cannot supervise it with a ground-truth state. We instead maintain

$$\widehat{S}_t(q) = \big\{(r_i, \mathcal{E}_i^+, \mathcal{E}_i^-, w_i)\big\}_{i=1}^K\,, \tag{30}$$

where r_i is a falsifiable preference claim, $\mathcal{E}_i^+$ and $\mathcal{E}_i^-$ are supporting and challenging records, and w_i is its current evidential weight. An atom names a decision aspect and states the preferred value.

Let $R_t^\star(q)$ be a finite, history-identifiable atomization of $S_t^\star(q)$: it contains only query-relevant factors whose values can, in principle, be supported or challenged by H_t. A proposer produces candidates $\widetilde{R}_t(q)$; an independent history verifier forms

$$B_t(q) = \Big\{r \in \widetilde{R}_t(q) : V_r(q, H_t) = \textsf{accept}\Big\}\,,$$

and a set gate returns

$$R_t(q) = G_q(B_t(q)).$$

Here $R_t^\star(q)$ plays the role of the valid set $\mathcal{F}^+(x)$ of Section 2; V_r stands for the grounding, support, and counterexample checks of Figure 2, and G_q for its merge and intent checks. Proposal therefore controls recall, whereas historical support and query-relevance control determine admission. Let $\mathfrak{R}$ be the finite candidate family, $\mathfrak{R}_q \subseteq \mathfrak{R}$ the atoms that can affect the decision posed by q, and define

$$d_\lambda(A, B) = \sum_{r \in A \triangle B} \lambda_r,$$

where $\lambda_r \geq 0$ measures the decision sensitivity of atom r and is distinct from the evidential weight w_i and from the memo-risk weights $a_{x,r}$ of Eq. 1. Let η_r^- upper-bound the end-to-end omission probability of each $r \in R_t^\star(q)$ and let η_r^+ upper-bound the admission probability of each $r \notin R_t^\star(q)$. Define

$$\epsilon_{\mathrm{rec}} = \sum_{r \in R_t^\star(q)} \lambda_r \eta_r^- + \sum_{r \in \mathfrak{R} \setminus R_t^\star(q)} \lambda_r \eta_r^+, \quad \epsilon_q = \sum_{r \in \mathfrak{R} \setminus \mathfrak{R}_q} \eta_r^+. \tag{31}$$

For any information Z given to the downstream policy, whether the history H_t, an atom set $A \subseteq \mathfrak{R}$, or a memo, let $\mathcal{R}_Z^\circ$ be the Bayes risk of deciding from (Q_t, C_t, Z); for a random recovered set, $\mathcal{R}_{R_t(q)}^\circ$ averages over proposal, verification, and gating randomness.

Proposition B.2 (Verifier-backed atomic recovery)**.** *The proposal–verification–gate procedure satisfies*

$$\Pr[R_t(q) \not\subseteq \mathfrak{R}_q] \leq \epsilon_q, \qquad \mathbb{E}[d_\lambda(R_t(q), R_t^\star(q))] \leq \epsilon_{\mathrm{rec}}.$$

If the ideal atomization satisfies

$$\mathcal{R}_{R_t^\star(q)}^\circ - \mathcal{R}_{H_t}^\circ \leq \epsilon_{\mathrm{id}},$$

and oracle risk is atomically stable,

$$\mathcal{R}_A^\circ - \mathcal{R}_{R_t^\star(q)}^\circ \leq d_\lambda(A, R_t^\star(q)) \quad \textit{for every } A \subseteq \mathfrak{R},$$

then

$$\mathcal{R}_{R_t(q)}^\circ - \mathcal{R}_{H_t}^\circ \leq \epsilon_{\mathrm{id}} + \epsilon_{\mathrm{rec}}. \tag{32}$$

We give explicit sufficient conditions for the end-to-end rates η_r^- and η_r^+ used in Proposition B.2. Throughout the proof, suppress (t, q) and write

$$R^\star = R_t^\star(q), \qquad \widetilde{R} = \widetilde{R}_t(q), \qquad B = B_t(q), \qquad R = R_t(q).$$

The finite candidate family is partitioned as

$$\mathfrak{R} = R^\star \,\dot\cup\, (\mathfrak{R}_q \setminus R^\star) \,\dot\cup\, (\mathfrak{R} \setminus \mathfrak{R}_q).$$

The three parts contain valid query-relevant atoms, invalid but query-relevant hypotheses, and query-irrelevant hypotheses, respectively.

We assume that the set gate is a selector and canonicalizer: modulo semantic canonicalization, it does not introduce a new preference claim that was absent from B.

Stagewise error conditions. For every valid atom $r \in R^\star$, suppose

$$\Pr[r \notin \widetilde{R}] \leq \delta_r^{\mathrm{prop}}, \tag{33}$$

$$\Pr[V_r \neq \textsf{accept} \mid r \in \widetilde{R}] \leq \beta_r, \tag{34}$$

$$\Pr[r \notin G_q(B) \mid r \in B] \leq \zeta_r^-. \tag{35}$$

These terms represent proposal omission, verifier false rejection, and set-gate removal of a valid atom.

For every invalid but query-relevant candidate $r \in \mathfrak{R}_q \setminus R^\star$, suppose

$$\Pr[V_r = \textsf{accept} \mid r \in \widetilde{R}] \leq \alpha_r. \tag{36}$$

For every query-irrelevant candidate $r \in \mathfrak{R} \setminus \mathfrak{R}_q$, suppose

$$\Pr[r \in G_q(B) \mid r \in B] \leq \zeta_r^+. \tag{37}$$

No independence between candidates or between the three stages is required.

One sufficient choice of the end-to-end bounds in Eq. 31 is therefore

$$\eta_r^- = \delta_r^{\text{prop}} + \beta_r + \zeta_r^- \qquad \text{for } r \in R^\star, \tag{38}$$

and

$$\eta_r^+ = \begin{cases} \alpha_r, & r \in \mathfrak{R}_q \setminus R^\star, \\ \zeta_r^+, & r \in \mathfrak{R} \setminus \mathfrak{R}_q. \end{cases} \tag{39}$$

Proof. Fix a valid atom $r \in R^\star$. The event $r \notin R$ can occur only if at least one of the following events occurs:

$$r \notin \widetilde{R}, \qquad r \in \widetilde{R} \text{ and } V_r \neq \textsf{accept}, \qquad r \in B \text{ and } r \notin G_q(B).$$

A union bound gives

$$\begin{aligned} \Pr[r \notin R] \leq \Pr[r \notin \widetilde{R}] \\ + \Pr[r \in \widetilde{R}, V_r \neq \textsf{accept}] \\ + \Pr[r \in B, r \notin G_q(B)]. \end{aligned} \tag{40}$$

Using Eqs. 33–35,

$$\Pr[r \notin R] \leq \delta_r^{\text{prop}} + \beta_r + \zeta_r^-. \tag{41}$$

Now fix $r \in \mathfrak{R}_q \setminus R^\star$. Because the set gate does not create new claims, $r \in R$ implies that $r \in \widetilde{R}$ and that the verifier accepted it. Consequently,

$$\begin{aligned} \Pr[r \in R] &\leq \Pr[r \in \widetilde{R}, V_r = \textsf{accept}] \\ &= \Pr[r \in \widetilde{R}] \, \Pr[V_r = \textsf{accept} \mid r \in \widetilde{R}] \\ &\leq \alpha_r. \end{aligned} \tag{42}$$

Finally, fix $r \in \mathfrak{R} \setminus \mathfrak{R}_q$. Admission requires the query gate to retain the atom, and hence

$$\begin{aligned} \Pr[r \in R] &= \Pr[r \in B] \, \Pr[r \in G_q(B) \mid r \in B] \\ &\leq \zeta_r^+. \end{aligned} \tag{43}$$

Equations 41–43 establish Eqs. 38 and 39.

For query relevance, observe that

$$R \not\subseteq \mathfrak{R}_q$$

if and only if at least one atom in $\mathfrak{R} \setminus \mathfrak{R}_q$ is admitted. Therefore,

$$\begin{aligned} \Pr[R \not\subseteq \mathfrak{R}_q] &= \Pr[\exists r \in \mathfrak{R} \setminus \mathfrak{R}_q : r \in R] \\ &\leq \sum_{r \in \mathfrak{R} \setminus \mathfrak{R}_q} \Pr[r \in R] \\ &\leq \sum_{r \in \mathfrak{R} \setminus \mathfrak{R}_q} \eta_r^+ \\ &= \epsilon_q. \end{aligned} \tag{44}$$

Next, expand the weighted symmetric difference:

$$\begin{aligned} \mathbb{E}[d_\lambda(R, R^\star)] = \sum_{r \in R^\star} \lambda_r \Pr[r \notin R] \\ + \sum_{r \in \mathfrak{R} \setminus R^\star} \lambda_r \Pr[r \in R]. \end{aligned} \tag{45}$$

Applying the end-to-end bounds gives

$$\begin{aligned} \mathbb{E}[d_\lambda(R, R^\star)] &\leq \sum_{r \in R^\star} \lambda_r \eta_r^- + \sum_{r \in \mathfrak{R} \setminus R^\star} \lambda_r \eta_r^+ \\ &= \epsilon_{\text{rec}}. \end{aligned} \tag{46}$$

For a random recovered set, define the oracle risk by averaging over estimator randomness:

$$\mathcal{R}^{\circ}_{R} = \mathbb{E}_{R}[\mathcal{R}^{\circ}(R)].$$

Atomic stability implies, for every realization of R,

$$\mathcal{R}^{\circ}(R) - \mathcal{R}^{\circ}_{R^\star} \le d_{\lambda}(R, R^\star).$$

Taking expectations and applying Eq. 46,

$$\mathcal{R}^{\circ}_{R} - \mathcal{R}^{\circ}_{R^\star} \le \epsilon_{\mathrm{rec}}. \tag{47}$$

By the ideal-atomization assumption,

$$\mathcal{R}^{\circ}_{R^\star} - \mathcal{R}^{\circ}_{H} \le \epsilon_{\mathrm{id}}. \tag{48}$$

Adding Eqs. 47 and 48 yields

$$\mathcal{R}^{\circ}_{R} - \mathcal{R}^{\circ}_{H} \le \epsilon_{\mathrm{rec}} + \epsilon_{\mathrm{id}},$$

which proves Proposition B.2. □

A sufficient condition for atomic stability. The stability assumption is implied by a bounded single-atom effect. Suppose that, for every intermediate atom set $A \subseteq \mathfrak{R}$ and every $r \in \mathfrak{R}$,

$$\left|\mathcal{R}^{\circ}_{A\triangle\{r\}} - \mathcal{R}^{\circ}_{A}\right| \le \lambda_r. \tag{49}$$

For any A, enumerate the atoms in $A\triangle R^\star$ as $r_1, \ldots, r_m$, set $A_0 = R^\star$, and obtain $A_j = A_{j-1}\triangle\{r_j\}$. Then $A_m = A$, and a telescoping argument gives

$$\begin{aligned}
\mathcal{R}^{\circ}_{A} - \mathcal{R}^{\circ}_{R^\star} &= \sum_{j=1}^{m} \left(\mathcal{R}^{\circ}_{A_j} - \mathcal{R}^{\circ}_{A_{j-1}}\right) \\
&\le \sum_{j=1}^{m} \lambda_{r_j} \\
&= d_{\lambda}(A, R^\star).
\end{aligned} \tag{50}$$

Thus, λ_r can be interpreted as a uniform upper bound on the decision-risk sensitivity to omitting or spuriously inserting atom r.

A signed-witness history verifier. One stylized verifier makes the error rates above concrete.

Conditional on a candidate r being proposed, suppose the history path extracts n_r qualified evidence units

$$\xi_{r,1}, \ldots, \xi_{r,n_r} \in [-1, 1].$$

Positive values encode explicit support for the proposed aspect value, negative values encode explicit incompatible evidence, and zero encodes weak, unrelated, or non-observing records. Define

$$\bar{\xi}_r = \frac{1}{n_r} \sum_{j=1}^{n_r} \xi_{r,j},$$

and accept only when

$$\bar{\xi}_r \ge \gamma_r, \qquad \gamma_r > 0.$$

If no qualified record is observed, the verifier abstains rather than treating absence as contradiction.

Assume the evidence units are conditionally independent for this stylized analysis. For a history-identifiable valid atom, suppose

$$\mathbb{E}[\xi_{r,j} \mid r \in R^\star] \ge \mu_r > \gamma_r.$$

Hoeffding's inequality then gives

$$\begin{aligned}
\beta_r &= \Pr[\bar{\xi}_r < \gamma_r \mid r \in R^\star] \\
&\le \exp\left(-\frac{n_r(\mu_r - \gamma_r)^2}{2}\right).
\end{aligned} \tag{51}$$

For an invalid query-relevant hypothesis, suppose instead

$$\mathbb{E}[\xi_{r,j} \mid r \notin R^\star] \le 0.$$

Then

$$\begin{aligned}\alpha_r &= \Pr[\overline{\xi}_r \ge \gamma_r \mid r \notin R^\star] \\ &\le \exp\left(-\frac{n_r \gamma_r^2}{2}\right).\end{aligned} \tag{52}$$

These bounds formalize two aspects of the implemented design. First, proposal recall remains necessary: a valid preference cannot be verified unless some discovery route proposes it. Second, explicit support and counterevidence make verifier precision improve as qualified evidence accumulates. On the gate's side, missing relevant atoms are already charged through η_r^- and ϵ_{rec}, and retained irrelevant atoms are controlled by ϵ_q.

Held-out temporal calibration. Post-query process and purchase evidence is not used to construct $\widetilde{R}$, run V_r, or apply G_q. It may subsequently calibrate w_i: explicit semantic agreement provides conservative temporal corroboration, an explicitly incompatible value provides counterevidence, and missing, weak, unrelated, or mixed observations produce abstention.

B.3 From Atomic Recovery to the Memo Bottleneck

For $Z \in \{H, M\}$, the history or the memo, let $\widehat{f}_Z$ be the downstream policy actually used with that input (the frozen π_ψ of Section 2 when $Z = M$) and $E_Z = \mathcal{R}_Z(\widehat{f}_Z) - \mathcal{R}_Z^\circ$ its excess risk over $\mathcal{R}_Z^\circ$; the subscript t is dropped below. Let ϵ_{syn} bound the oracle-risk gap incurred when the memo expresses the recovered preference information:

$$\mathcal{R}_M^\circ - \mathcal{R}_{R(Q)}^\circ \le \epsilon_{\mathrm{syn}}. \tag{53}$$

Combining Eq. 53 with Proposition B.2 gives

$$\mathcal{R}_M^\circ - \mathcal{R}_H^\circ \le \epsilon_{\mathrm{id}} + \epsilon_{\mathrm{rec}} + \epsilon_{\mathrm{syn}} =: \epsilon_{\mathrm{comp}}. \tag{54}$$

For a faithful, decodable serialization of the recovered statements, one may take $\epsilon_{\mathrm{syn}} = 0$; a learned free-form memo pays omissions, the coverage risk of Eq. 1, and unsupported or distorted claims, its faithfulness risk, through this term.

Since $\mathcal{R}_Z(\widehat{f}_Z) = \mathcal{R}_Z^\circ + E_Z$,

$$\begin{aligned}&\mathcal{R}_M(\widehat{f}_M) - \mathcal{R}_H(\widehat{f}_H) \\ &\quad = (\mathcal{R}_M^\circ - \mathcal{R}_H^\circ) + (E_M - E_H) \\ &\quad \le \epsilon_{\mathrm{comp}} - (E_H - E_M).\end{aligned} \tag{55}$$

Therefore,

$$E_H - E_M > \epsilon_{\mathrm{comp}}$$

is sufficient for

$$\mathcal{R}_M(\widehat{f}_M) < \mathcal{R}_H(\widehat{f}_H).$$

The deployed memo policy of Section 2 does not receive R: it writes the memo directly from (Q, H), so its ϵ_{syn} measures amortization and expression error relative to the verified atoms, and with a faithful expression the verified atoms themselves are the bottleneck. A history-copying memo may have a small oracle-information gap, but it generally does not reduce the finite-model burden $E_H - E_M$; the bound accordingly gives it no automatic advantage. RQ1 compares the two sides of Eq. 55 downstream, and Appendix C.5 compares the two sides across downstream policies.

B.4 Task-Switch Gap at θ_P

This subsection explains why the checkpoint θ_P right after a proposer phase can write worse memos than θ_k even when the proposer phase has improved the shared representation, the pattern behind the

coverage dip at θ_P in Section 4.3 and its recovery in Figure 3d. Locally write the shared checkpoint as $\theta = (\phi, \omega_M, \omega_P)$, where ϕ parameterizes shared features and ω_M, ω_P are effective role-specific readouts. Let $\mathcal{L}_M(\theta)$ be the memo objective and

$$\bar{\mathcal{L}}_M(\phi) = \inf_{\omega_M} \mathcal{L}_M(\phi, \omega_M) \tag{56}$$

the best memo loss recoverable after readapting the memo readout. With the checkpoints of Eq. 6, let ϕ_0 and ϕ_J be the shared features of θ_k and θ_P, and define

$$\begin{aligned} \Delta_0 &= \mathcal{L}_M(\theta_k) - \bar{\mathcal{L}}_M(\phi_0) \geq 0, \quad \Delta_{\text{switch}} = \mathbb{E}\big[\mathcal{L}_M(\theta_P) - \bar{\mathcal{L}}_M(\phi_J)\big] \geq 0, \\ G_{\text{rep}} &= \bar{\mathcal{L}}_M(\phi_0) - \mathbb{E}\big[\bar{\mathcal{L}}_M(\phi_J)\big], \end{aligned} \tag{57}$$

the initial memo optimization gap, the post-proposer task-switch gap of the memo readout, and the gain in recoverable memo performance; the first two are nonnegative because the profiled objective is an infimum over memo readouts. Substituting the definitions gives the exact decomposition

$$\mathbb{E}\big[\mathcal{L}_M(\theta_P)\big] - \mathcal{L}_M(\theta_k) = \Delta_{\text{switch}} - G_{\text{rep}} - \Delta_0. \tag{58}$$

Proposer training may therefore improve the representation, $G_{\text{rep}} > 0$, while worsening immediate memo performance whenever $\Delta_{\text{switch}} > G_{\text{rep}} + \Delta_0$: θ_P writes worse memos because its memo readout is misadapted, not necessarily because the preference representation has become less useful. If the memo phase readapts the readout to within ϵ_{adapt} of the profiled objective, $\mathbb{E}[\mathcal{L}_M(\theta_{PM}) - \bar{\mathcal{L}}_M(\phi_J)] \leq \epsilon_{\text{adapt}}$, then

$$\mathbb{E}\big[\mathcal{L}_M(\theta_{PM})\big] - \mathcal{L}_M(\theta_k) \leq -G_{\text{rep}} + \epsilon_{\text{adapt}} - \Delta_0, \tag{59}$$

so the recovered checkpoint improves over θ_k whenever $G_{\text{rep}} > \epsilon_{\text{adapt}} - \Delta_0$, even if θ_P is worse than θ_k. The comparison with route D at matched updates is Theorem A.4.

C Implementation Details of PREMO

This appendix records the data, evaluation protocol, additional analyses, and implementation details behind Sections 3 and 4.

C.1 Tasks and Data

Table 5 summarizes the three tasks of Section 4.1; all splits are by user, and every model is trained and evaluated within one task.

APeB. We use the benchmark's sessions and hard candidate lists (Yang et al., 2026). GPT-5.5 and Gemini 3.6 Flash score how well each session's purchase matches its query, and we keep the top 30% by the averaged score (a manual audit of 100 kept sessions accepted 98%). The benchmark's queries are shopping intents, but a vague one such as "outfit" leaves the purchase only loosely tied to the query; the filter keeps the sessions whose purchase matches the stated intent. It is applied once, before the split, and every method uses the same sessions, so all APeB results are on this intent-matched subset. This leaves 1,996 users with one query–history pair each: 1,594 for training, 201 for validation, and 201 for testing. Training uses no candidate choice; each validation and test pair is asked five times, with the purchased item and a fresh uniform draw of four hard candidates, and Hit@1 is averaged over the draws. The candidates and the purchase are post-query behavior: during training only the weighting step of Section 3.1 sees them, and at evaluation only the downstream policy sees the candidates. H lists the user's earlier sessions in the benchmark's format, product titles and feature descriptions and the titles and captions of watched videos, with at most 120 records (mean 118.8), 22.1k characters, and 6.3k tokens (Qwen3.5-4B tokenizer, history records only).

PersonaMovieLens. MovieLens 32M (Harper & Konstan, 2015) has no queries, so we synthesize them. For each user we sort the ratings by time and take as target Y a movie the user later rated 5.0, with at least 240 earlier ratings and 30 later events; H keeps the 120 most recent earlier ratings. Each movie is shown with title, year, genres, tags, the user's rating, and, when a verified profile exists, a plot summary, director, up to three lead actors, themes, mood, setting, and keywords; profiles are written by GPT-5.5 from IMDb (IMDb, 2026) and Wikipedia evidence and verified before use. The

candidates are the target plus 14 movies of the same genre family, chosen against the target's genres and tags and the user's history. GPT-5.5 writes the query, a broad viewing request of 8–16 words, from the genre family, a generic genre description, preset mood and theme words, and the number of candidates, without the target's title, year, rating, tags, or synopsis or the user's history. Queries are checked for length, banned words, and title leakage, and a GPT-5.5 audit with web evidence keeps only queries that fit the target. Of 9,220 initial cases, 8,000 pass the format checks, 5,297 the fit audit, and 4,668 the profile-quality filter (target synopsis verified, at most 15% of candidate synopses missing); one case per user gives 2,500 training, 900 validation, and 924 test users. Evaluation is a five-way choice, the target plus four of the 14 candidates drawn at random. Histories average 67.9k characters and 18.0k tokens (Qwen3.5-4B tokenizer, history fields only).

PersonaMem-v3. From the released query set of Jiang et al. (2026) we keep the `chatbot_personalized_response` task type, whose queries ask for a reply conditioned on the user's preferences, and drop the recommendation-reranking, over-personalization, agentic, and proactive families. Of its 1,256 queries, 5 lack a supporting history, leaving 1,251 over 99 users (the benchmark's personas), split by user into 1,001 training, 125 validation, and 125 test queries. H is the user's time-indexed engagement history across the benchmark's platforms (social media, chatbot, AI companion, calendar), cut at the query timestamp. The raw history averages about 400k tokens, so we compact it before any policy sees it: event fields are trimmed, non-user conversation turns dropped, records kept in priority order (privacy boundaries, corrections, then records relevant to the query) up to 120 records and 40k characters per query, and sorted by time. Compacted histories average 119.97 records, 34.0k characters, and 10.5k tokens (Qwen3.5-4B tokenizer, history only). The compaction is query-dependent, and all methods, including the full-history baseline, read the same compacted history. There is no candidate set ($C = \emptyset$): the downstream policy writes the reply from (Q, M), and GPT-5.6 Sol, as a frozen judge, scores it on three 0–10 components. An evidence builder first verifies the benchmark's annotated target and distractor preferences for the query against the history; with this evidence and the golden response Y the judge scores task completion (correct, feasible, fulfills the request), preference alignment (uses history-supported, permitted preferences), and rule compliance (respects stated boundaries and hard constraints, fabricates no personal facts about the user or a third party). The Task score of Table 1 weights them 0.4, 0.3, and 0.3; we report it on a 0–100 scale, ten times the benchmark's native 0–10 score, so that it reads like the other columns.

Post-query behavior. Only APeB has post-query behavior (the target session's later browsing and purchases), which sets the atom weights of Section 3.1; on PersonaMovieLens and PersonaMem-v3 the weights come from the support strength of the cited records (Appendix C.10).

Table 5: Tasks. Train and Test count instances; $|H|$ is the mean history length in tokens (Qwen3.5-4B tokenizer); $|C|$ is the candidate-set size.

Task	History content	Goal	Users	Train	Test	$\lvert H\rvert$	$\lvert C\rvert$
APeB	sessions, purchases	bought product	1,996	1,594	201×5	6.3k	5
PersonaMovieLens	ratings + profiles	later 5-star movie	4,324	2,500	924	18.0k	5
PersonaMem-v3	cross-platform events	personalized reply	99	1,001	125	10.5k	–

C.2 Evaluation Protocol

Evaluation atoms. For each test instance, $E(x)$ is the set of atoms GPT-5.4 proposes over four rounds through the verification path of Appendix C.10, with GPT-5.4 also running the checks, capped at 20 accepted atoms with their weights and frozen before evaluation. It is also the reference pool of RQ3's frontier recall; the starting inventory A_0 of Section 3.4 is its first round. Since GPT-5.4 proposes $E(x)$, the coverage, faithfulness, and grounding of GPT-5.4's own memos in Table 1 are scored against atoms it proposed itself. Appendix C.3 audits $E(x)$ and the judge.

Memo scoring. GPT-5.5 is the frozen judge for all three memo scores, with prompts that differ from training. Coverage: the judge scores each atom of $E(x)$ as 0, 0.5, or 1 (absent, partial, full), and coverage is the mean weighted by the weights $E(x)$ carries from its own verification run, on

every task. Faithfulness: the judge reads the query, the memo, and the full history, and gives one 0–1 score for unsupported or overconfident statements; faithfulness is one minus this score. Grounding: the memo is split into sentence units, the judge splits each unit into atomic claims and labels each supported, partial, or unsupported by whether the records the claim cites or relies on exist in the history and support it; grounding is the number of supported claims plus half the partial ones over all claims. Two annotators accepted 95%, 98%, and 97% of the judge's coverage, faithfulness, and grounding verdicts on 100 audited samples per metric (Appendix C.3). On PersonaMem-v3, faithfulness follows the benchmark's constraint semantics and averages three scores over the 125 test memos: boundary compliance, the share of memos with no clear violation of a forget, do-not-personalize, retraction, or re-authorization event in the visible history (uncertain cases do not count); average restraint compliance, one minus the mean over memos of the mean violation score of the memo's frozen personalization-restraint rubrics, each scored 0, 0.5, or 1 on the whole memo; and the violation-free rate, the share of memos with all restraint scores 0.

Summary baselines. Persona: six persona claims written from the full history, at most 1,200 characters. P-Check (Seo & Lee, 2026), following the original method: a general preference of at most 1,200 characters is written from the full history alone and serves as the memo; a checklist generator maps it and the query to candidate-agnostic criteria labelled Essential, Important, or Optional by preference-contrastive criterion weighting, and for ranking the downstream policy receives the general preference, the query, and this checklist. The generator is Qwen3.5-4B fine-tuned by next-token prediction on each task's training split (full-parameter text model, vision module frozen, 3 epochs, peak learning rate 2×10^{-5}, two H100 GPUs, best validation checkpoint); the paper's criterion-weighted judge reward is not used. Mem0: all final memories after the persisted store is reopened, as a JSON array sorted by id. Persona and Mem0 are produced by the untrained Qwen3.5-4B (Mem0 uses it as its extraction LLM); all three baselines cite history records, which their grounding scores check.

Downstream policies. Every reported number is the mean over three seeds; a seed fixes the sampling of the memo policy and the decoding of the downstream policy from one trained checkpoint, and no model is retrained per seed. APeB's Hit@1 is further averaged over the five candidate draws of Appendix C.1 within each user, so the user, not the draw, is the unit of the intervals in Appendix C.4. Condensed prompts are given in Appendix E.

C.3 Reliability of the Evaluation

Reference pool audit. Coverage is measured against $E(x)$, a verified reference pool. To check the pool, two annotators independently reviewed the $E(x)$ of the same 100 APeB test users and judged, for every atom, whether the history supports it and whether it applies to the query. Both annotators accepted 91% of the atoms. The atoms that at least one annotator rejected are mostly over-inferences: preferences that reach further than the history warrants, which the verifiers had let through. Coverage numbers therefore include a small share of contested atoms.

Judge audit. The same two annotators reviewed 100 judge outputs per metric on PersonaMem-v3. For coverage they checked the per-atom labels, counting a label one step away (0.5 against 1) as agreement; for faithfulness and grounding they read the score together with its rationale and accepted or rejected the verdict as a whole. Both annotators accepted 95%, 98%, and 97% of the coverage, faithfulness, and grounding verdicts. Coverage disagreements are rare because the presence of an atom in a memo is easy to see; the rejected faithfulness verdicts are cases where the rationale does not fully justify the score; the rejected grounding verdicts are memos whose citations are too generic to pin to a record.

Dependence on the reference pool. Two checks ask whether the comparison between methods depends on who built the pool. First, on 100 APeB test instances where the judge scored the coverage of the PREMO memo above that of the GPT-5.4 memo, the two annotators compared the two memos directly against the history; both agreed with the judge's ordering on 92% of the instances. Second, on the same 100 instances we rebuilt the reference pool with Gemini 3.6 Flash in place of GPT-5.4 and rescored both memos; the ordering held on 97%. The coverage claims of Section 4 are claims about coverage of the verified reference pool.

C.4 BOOTSTRAP INTERVALS

Table 6 gives 95% intervals for the downstream differences of Table 1 between PREMO and the two closest rows, direct memo training D and the GPT-5.4 memo, on the same resampling units as below: users on APeB, with scores averaged over the three seeds and the five candidate draws within each user; users on PersonaMovieLens, with scores averaged over the three seeds; and personas on PersonaMem-v3. PREMO is ahead of D on APeB and PersonaMem-v3 and ahead of the GPT-5.4 memo on APeB with intervals that exclude zero; the other three differences include zero, and we read them as comparable performance.

Table 6: Downstream differences of Table 1 (PREMO minus the named row) with 95% bootstrap intervals; Hit@1 on APeB and PersonaMovieLens, task completion score on PersonaMem-v3.

Task	PREMO $-$ D	PREMO $-$ GPT-5.4 memo
APeB	$+3.7$ [1.1, 6.1]	$+4.4$ [1.3, 7.5]
PersonaMovieLens	$+0.9$ [-0.8, 3.0]	$+0.1$ [-1.8, 2.0]
PersonaMem-v3	$+2.1$ [0.2, 4.4]	$+0.8$ [-1.4, 3.0]

For Table 2, Table 7 repeats the APeB side of Table 2 with 95% intervals for the paired difference between the PREMO memo and the GPT-5.4 memo under each downstream policy. Scores are first averaged over the three seeds and the five candidate draws within each test user; the intervals are percentile bootstrap intervals over the 201 test users with 20,000 resamples. We also report NDCG@5, which scores the rank of the purchased item among the five candidates: with one relevant item, NDCG@5 $= 1/\log_2(1 + \text{rank})$, so it is 1 when the item is ranked first and 0.39 when it is ranked last. Every APeB difference excludes zero. Table 8 does the same for the PersonaMem-v3 side with the task completion score and its three components, paired by query and bootstrapped with personas as groups (50,000 resamples). All total differences include zero.

Table 7: APeB test set: Hit@1 and NDCG@5 of the GPT-5.4 memo and the PREMO memo under five downstream policies, with the difference of the means shown and its 95% bootstrap interval over the 201 test users.

	GPT-5.4 memo		PREMO memo		Difference [95% CI]	
Downstream	Hit@1	NDCG	Hit@1	NDCG	Hit@1	NDCG
Qwen3.5-4B	61.5	0.830	65.9	0.851	$+4.4$ [1.3, 7.5]	$+0.021$ [0.006, 0.036]
Gemini 3.6 Flash	67.4	0.863	72.8	0.887	$+5.4$ [2.5, 8.7]	$+0.024$ [0.011, 0.038]
GPT-5.4	66.6	0.858	73.4	0.888	$+6.8$ [3.7, 10.2]	$+0.030$ [0.016, 0.044]
GPT-5.5	69.4	0.868	72.4	0.884	$+3.0$ [0.2, 6.1]	$+0.016$ [0.002, 0.029]
GPT-5.6 Luna	67.0	0.859	70.2	0.874	$+3.2$ [0.2, 6.2]	$+0.015$ [0.002, 0.027]

Table 8: PersonaMem-v3 test set: task completion score (0–100) of the GPT-5.4 memo and the PREMO memo under five downstream policies, the difference of the means shown, and the bootstrap differences of its three components (task completion, preference benefit, rule compliance), each with a 95% interval over the 125 test queries grouped by persona.

	Score		Difference [95% CI]			
Downstream	GPT-5.4	PREMO	Total	Completion	Preference	Compliance
Qwen3.5-4B	82.5	83.3	$+0.8$ [-1.4, 3.0]	$+1.6$ [-0.8, 3.7]	-0.5 [-2.6, 1.6]	$+0.9$ [-1.9, 4.6]
Gemini 3.6 Flash	91.1	91.0	-0.1 [-1.6, 1.4]	0.0 [-1.2, 1.2]	0.0 [-1.6, 1.9]	-0.5 [-2.3, 1.4]
GPT-5.4	92.0	92.7	$+0.7$ [-0.2, 2.2]	$+1.7$ [0.8, 2.8]	$+0.6$ [-1.1, 2.7]	-0.3 [-1.8, 1.4]
GPT-5.5	92.8	92.5	-0.3 [-1.5, 1.1]	-0.2 [-1.0, 0.7]	-0.9 [-2.8, 1.3]	0.0 [-1.2, 1.5]
GPT-5.6 Luna	91.8	92.0	$+0.2$ [-0.7, 1.4]	$+1.2$ [0.3, 2.2]	0.0 [-1.4, 1.5]	-0.8 [-2.8, 1.2]

C.5 A MEMO REPLACES THE FULL HISTORY ACROSS DOWNSTREAM POLICIES

On PersonaMem-v3 we give each downstream policy the query alone, the query with the full 120-record history, or the query with a memo that the same policy writes from that history before answering, with the memo prompt of Section 3.3 and no training (Table 9). For Qwen3.5-4B the history

lowers the score below the query alone, and its own untrained memo lowers it further; only the trained memo policy of Table 1 turns the history into a gain. For Qwen3.8-27B and Qwen3.5-35B-A3B the three inputs are within 2.5 points. For the frontier policies the history helps by 6 to 8 points, and a memo of about 170 tokens matches or exceeds it for three of the four. A memo thus carries what the history contributes to the decision at a small fraction of its length, and the loss from reading the history directly is largest for the smallest policy, the excess-risk term E_H of Appendix B.3.

Table 9: PersonaMem-v3 test set, task completion score (0–100) of each downstream policy reading the query alone, the full 120-record history, or a memo it writes itself from that history. Qwen3.5-4B is the three-seed mean of Table 1; the frontier policies run at high reasoning effort.

Downstream policy	Query only	Full history	Own memo
Qwen3.5-4B	65.5	63.5	61.5
Qwen3.5-35B-A3B	80.2	80.4	79.6
Qwen3.8-27B	79.6	82.0	81.8
GPT-5.4	84.1	90.6	91.9
GPT-5.6 Luna	83.4	91.1	91.9
Gemini 3.6 Flash	83.1	90.3	90.8
Gemini 3.7 Flash	84.0	90.9	89.9

C.6 What SFT Learns Versus What Role Alternation Learns

We compare the memos of the SFT policy of Table 1 (after memo SFT) and of the PREMO policy on 100 APeB test instances with four lexical measures against the evaluation atoms and the history (Table 10): the share of atom tokens that appear in the memo, the share of atoms with a lexical match in the memo, the share of preference statements that name specific history evidence, and the share of memos that state which preference matters more. SFT supplies the behavioral prior: from GPT-5.4's demonstrations the model learns to compress the history into a memo without seeing the candidates, and its memos already match three quarters of the atoms and state a trade-off in most cases. Alternation adds task-specific selection pressure: the weighted memo reward pays for covering verified atoms, so the trained memos match more of them, ground more of their statements in specific history, and state priorities more often, giving concrete budget tiers, finishes, sensitivities, compatibilities, and conditions to avoid. The downstream policy thus receives a memo that says what to weigh, what to avoid, and what matters most. Table 11 shows one query. The SFT memo is fluent but hedges on nearly everything; a memo policy trained by GRPO from the base model without SFT cites many history items but reads as a list; the PREMO memo keeps the fluency, grounds each preference in the history, marks the inference where direct evidence is thin, and ends with the priority order.

Table 10: Lexical measures of memos on 100 APeB test instances, SFT policy versus PREMO policy.

	SFT	PREMO
Atom tokens appearing in the memo	0.40	**0.47**
Atoms with a lexical match in the memo	0.75	**0.81**
Preference statements naming specific history	0.49	**0.60**
Memos stating which preference matters more	0.84	**0.91**

C.7 What Later Rounds Discover

On APeB we compare the atoms the 4B proposer discovers in the first two rounds with those of the last two rounds by their post-query weight, sampling 100 positive-weight atoms ($w_r \geq 0.5$) from each. Among them, full-weight atoms, which the user's later behavior clearly supports, fall from 37 to 30 percent, and half-weight atoms rise from 63 to 70 percent. Later rounds thus still find atoms that post-query behavior supports, with a somewhat larger share supported only weakly.

Table 11: Condensed memos for the APeB query "shampoo recommendation" from the SFT policy, from a memo policy trained by GRPO from the base model without SFT, and from PREMO; product names and prices are removed.

SFT	GRPO without SFT	PREMO
The user is looking for a shampoo recommendation. The history holds little direct evidence about hair care, but the user browsed a volumizing curling wand, which suggests an interest in styling and in products that add volume or shape curls gently rather than hold or straighten strongly. Their shopping leans toward affordable everyday items and practical personal-care products, so an affordable shampoo is probably important. They also show some interest in skincare and health products, so moisturizing or scalp-friendly claims may matter, though this is uncertain. Overall the user probably prefers a practical, reasonably priced shampoo with some hair-care benefit, but the history does not settle hair type, scent, or ingredient preferences.	Based on recent behavior, the user may be looking for a gentle, affordable shampoo, possibly free of sulfates or preservatives. Evidence includes interest in a styling clip [...], several low-cost conditioners and makeup sets, and a low-cost hair-removal kit. The history shows a clear preference for drugstore brands and at-home beauty: adding cheap supplements to the cart, watching videos of budget skincare brands [...], and frequently buying small low-cost beauty accessories. Rather than premium shampoos, the user seems to care about effective, gentle, non-damaging formulas, and may value volume or curl care like the styling tool they browsed. Consider products under [...] with natural ingredients or "damage-free" claims, which fits the user's value-for-money home beauty pattern.	The user is looking for a shampoo recommendation. From the history they likely prefer affordable everyday personal-care items over premium brands, shown by repeated attention to budget skincare, sunscreen, toner and moisturizer sets, fragrance, digestive aids, probiotics, hair-removal tools, and medicated patches. Their beauty behavior suggests they value gentle, practical, budget-friendly hair care for daily use, with added benefits such as volume, curl protection, or moisture rather than salon-grade treatment. They also seem to favor value bundles and multipacks, so a low-priced, easy-to-use shampoo that fits a simple routine may appeal most. Because the history holds little direct shampoo browsing, these preferences are inferred from broader personal-care behavior; the user most likely prioritizes low price, everyday convenience, and basic hair-care benefits over specialized or premium formulas.

C.8 Additional Analyses for RQ2

Training curves of Figure 3. Each panel plots one GRPO phase against its optimizer steps, using the metrics that the training verifiers (Appendix C.10) compute on the training prompts of that phase. Coverage is measured against the current inventory A_k. Panel (a) is route D's first memo phase at 4B, started directly from the post-warm-up checkpoint; the plot shows the first 200 of its 500 steps. Panels (b) and (c) are route A's first proposer phase from the same checkpoint, at 4B and 35B-A3B. Panel (d) is route A's final memo phase at 35B-A3B, started from θ_P; the dashed line is the coverage that route D reaches at the end of its final memo phase, measured the same way.

Penalty weight does not fix memo-only training. We rerun the first memo GRPO phase of route D on PersonaMem-v3, from the post-warm-up checkpoint, with $\lambda \in \{0.1, 0.5, 1, 2, 5\}$ in Eq. 5 (Table 12). Every λ raises coverage above the checkpoint and lowers faithfulness below it. A larger λ recovers part of the faithfulness, from 59.2 at $\lambda = 0.1$ to 78.4 at $\lambda = 5$, but stays 9 points below the checkpoint's 87.3, and at $\lambda = 5$ the coverage gain is gone and grounding falls. The judge scores only atoms in $I_k \cup \widehat{S}_k^-$ (Appendix C.11), so a claim outside both sets cannot be penalized at any λ.

Inventory effect versus weight transfer. A 2×2 factorial trains the memo role of the final 4B round on APeB from θ_4 or θ_P, on A_4 or $A_4 \cup N_4$, at equal memo budget (Table 13). The new atoms carry the coverage: from either checkpoint they add 1.4 and 0.5 points, and without them coverage stays at the previous round's 72.1. The proposer checkpoint carries faithfulness: it adds 6.7 and 7.4 points on the two inventories, with coverage nearly unchanged. Downstream accuracy needs both: either factor alone adds at most 0.4 points, their combination adds 2.2, an interaction of 2.0 points. Adding the atoms to the reward buys coverage, but the checkpoint that discovered them is the one that turns them into better decisions.

Table 12: Penalty weight ablation on the PersonaMem-v3 test set: one memo GRPO phase of route D from the post-warm-up checkpoint (after memo and proposer SFT) with different λ in Eq. 5.

	Cov.	Faith.	Ground.
Post-warm-up checkpoint	66.6	**87.3**	**62.9**
$\lambda = 0.1$	74.6	59.2	62.1
$\lambda = 0.5$	74.3	73.4	62.6
$\lambda = 1$ (default)	73.9	67.6	61.8
$\lambda = 2$	**74.8**	75.2	59.6
$\lambda = 5$	68.8	78.4	57.6

Table 13: Checkpoint $\times$ inventory factorial for the final 4B round on APeB at equal memo budget: coverage / faithfulness / Hit@1 of the trained memo policy; the interaction row is $[(\theta_P, A_4 \cup N_4) - (\theta_P, A_4)] - [(\theta_4, A_4 \cup N_4) - (\theta_4, A_4)]$.

Inventory	from θ_4	from θ_P
A_4	72.1 / 70.6 / 63.7	73.3 / 77.3 / 64.1
$A_4 \cup N_4$	73.5 / 69.7 / 63.5	73.8 / 77.1 / 65.9
interaction	-0.9 / $+0.7$ / $+2.0$	

C.9 Additional Analyses for RQ4

Training curves of Figure 5. Each curve is one memo GRPO phase of Qwen3.5-4B on APeB, started from the post-warm-up checkpoint and rewarded on an inventory built by the single judge, by the judge plus the support and counterexample checks, by the judge plus the merge and intent checks, or by the full path with post-query weights. Metrics are computed as in Figure 3 (Appendix C.8): by the training verifiers on training prompts, with coverage against the inventory of that run. Table 4 reports test-set metrics after the full PREMO training with each verifier.

A single judge is unstable. We sample 1,000 proposals from PersonaMem-v3 training and verify each one three times, by the single judge and by the full path, with independent calls. Averaged over the three calls, the single judge accepts 77.3 percent of the proposals and the path 52.1 percent. The single judge's verdict changes across the three calls for 37.4 percent of the proposals; the path's verdict changes for 12.3 percent.

What each verifier lets through. We audit 1,000 atoms accepted during PersonaMem-v3 training under each verifier variant of Table 4, judged by GPT-5.6 Luna on three criteria (Table 14). Without the support and counterexample checks, the share of accepted atoms fully supported by the history falls by 14.8 points and citation errors nearly double, from 13.6 to 26.8 percent; merge does not make up for the missing evidence check. Without merge, the support rate is the highest of all variants, but half of the accepted atoms add no new information, and this share rises from 28.6 percent in the first 40 training steps to 53.3 percent in the last 20: the proposer earns a growing reward for restating what the inventory already holds. The single judge accepts the fewest duplicates but, after the variant without the verifier, the most unsupported and mis-cited atoms; new is not the same as true. The full path is the most trustworthy but does not solve duplicate acceptance: a quarter of its accepted atoms add no new information. PREMO thus trains on supervision that still carries errors, and the gains of Section 4 are obtained under this error rate.

Table 14: Audit of atoms accepted during PersonaMem-v3 training under each verifier variant: 1,000 accepted proposals per variant, judged by GPT-5.6 Luna; columns as in Table 4.

Share of accepted atoms (%)	Path	Single	w/o Verif.	w/o Merge
Fully supported by the history (↑)	**86.2**	73.7	71.4	88.9
With a citation error (↓)	**13.6**	20.7	26.8	15.1
Adding no new information (↓)	24.8	**10.0**	28.8	50.3

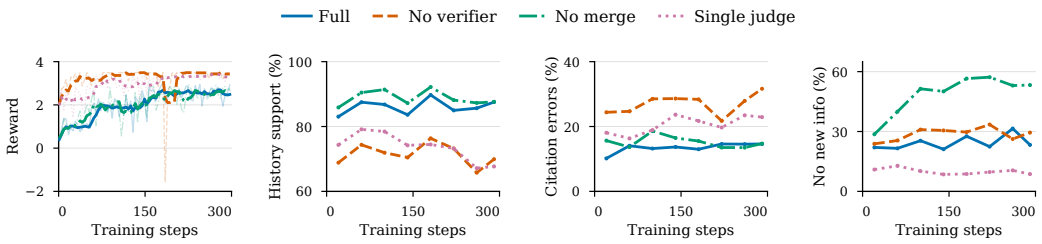


Figure 6: Proposer GRPO under each verifier variant on PersonaMem-v3 training: the proposer's training reward before normalization (the sum of slot scores of Eq. 4) and, for the atoms accepted at each step, the three audit criteria of Table 14. Without merge the reward keeps rising while the share of accepted atoms adding no new information climbs past half.

What each proposer learns to propose. After the first proposer GRPO phase under each verifier variant, we prompt the best checkpoint of that phase to propose atoms on the PersonaMem-v3 test set, showing it an inventory of five existing atoms, and verify its proposals with the full path (Table 15). All four propose about five candidates per instance, but the share the full path accepts differs: 55 percent for the proposer trained with the full path, 51 percent without merge, 30 percent with the single judge, and 22 percent without the support and counterexample checks. Merge shows up after acceptance: the proposer trained without it has 1.05 accepted atoms per instance removed as duplicates, against 0.49 for the full path, so it adds fewer new atoms than the single-judge proposer despite having more accepted. The verifier thus trains recognition and merge trains novelty; a proposer trained without either learns to propose atoms that the full path rejects or the inventory already holds.

Table 15: Proposals on the PersonaMem-v3 test set by the first-round proposer checkpoint trained under each verifier variant, given five existing atoms and verified by the full path: candidates, accepted atoms, and new atoms after MERGE, per instance; columns as in Table 4.

Per instance	Path	Single	w/o Verif.	w/o Merge
Candidates proposed	4.99	5.00	4.88	4.98
Accepted by the full path	**2.75**	1.52	1.06	2.52
New atoms after MERGE	**2.26**	1.45	0.97	1.47

What post-query weights teach. Table 4 shows that removing the post-query weights leaves coverage, faithfulness, and grounding within half a point but lowers Hit@1 by 2.3 points. The weights grade the training atoms by how the user's later behavior bears them out, and the trained policy grades its own claims the same way. We compare the memos of the two policies on 50 APeB test cases, selected to cover product categories, short and long histories, and both directions of the downstream difference. The memo written with weights separates the preferences the history establishes from those it only suggests: it states the former plainly with their evidence and the latter with hedged wording (likely, appear, seem, suggests, implies), which occurs 3.9 times per 100 words against 1.4 without weights. The memo written without weights states every preference with the same force and quotes the history as its evidence: 44 percent of its tokens also occur in the history against 33 percent with weights (lower in 49 of the 50 cases), and sentences that restate a history action (viewed, added, bought) occur 2.5 times per 100 words against 1.1. Table 16 shows one pair. The memo without weights lists product titles under a single "strongly signals"; the memo with weights states the established preference for low-cost, oil-based, travel-sized fragrances and marks the versatile-scent preference as possible.

C.10 VERIFICATION PATH

The path $\mathcal{V}$ runs the steps of Figure 2: GROUND, a deterministic citation check on the proposer's own grounding; SUPPORT and COUNTEREX, two model checks run in parallel; MERGE and INTENT, which update the inventory; and WEIGHT, which sets the atom weight; plus parsing, retry,

Table 16: Excerpts of the memos for the APeB query "Best fresh inspired perfume" from the policies trained without and with post-query weights; hedged wording in italics.

Path w/o Weight	Path
"History strongly signals a preference for budget-friendly, affordable, and oil-based or high-concentration fragrances (e.g., repeated views and purchases of oil-based dupe scents like Stranger with You, JPG Le Male-inspired, HSI Scents, Noir Scents [...]). They frequently explore student/professional/daily-use positioning, travel-friendly mini sizes, and 'fresh,' 'sweet,' 'floral,' 'powdery,' 'citrus,' and 'woody-fruity' notes (e.g., Hottie Lab Lemon Nocturne, HSI Scents woody fruity, Sellion Brown Sugar Fantasy [...])."	"[...] your consistent engagement with oil-based concentrates, small travel-sized bottles (25ml–50ml), and 'dupe' or 'inspired' branding indicates that you prioritize cost-effectiveness and long-lasting performance [...] over luxury packaging or brand prestige. You *seem* particularly interested in unisex or versatile scents suitable for daily wear, students, and professionals, *implying* a need for a fragrance that is subtle yet impactful enough for everyday activities."

failure, and capacity rules. During training, SUPPORT, COUNTEREX, MERGE, INTENT, and the memo judge all use GPT-5.6 Luna; when the initial inventory A_0 and the reference sets $E(x)$ are built, the same checks use GPT-5.4. The frontier models used anywhere in this paper are called through their APIs under the identifiers `gpt-5.4-2026-03-05`, `gpt-5.5`, `gpt-5.6-luna`, `gpt-5.6-sol`, `gemini-3.6-flash`, and `gemini-3.7-flash`; prompt versions are fingerprinted per run.

GROUND. Grounding is part of the proposer's output: for each question it writes a short answer and the strongest one to three `(index, quote)` anchors from the indexed history. The current query only delimits scope and is never evidence; neither are the candidate wording or the proposer's rationale. GROUND requires a non-empty answer and anchors whose quotes match the cited history lines after Unicode, case, whitespace, and punctuation normalization; failures are quarantined before any model-based verification.

SUPPORT. SUPPORT reads only the cited anchors and their full lines. Acceptance is enforced in code as `accept` = `history_supported` $\wedge$ `query_relevant` $\wedge$ `privacy_safe`: the answer must be directly supported or faithfully paraphrased at a strength matching the behavior (one purchase can support a narrow preference; browsing supports only "may be interested"), and the preference must belong to the current product and search context, and the atom must not use revoked or do-not-personalize information. There is no minimum evidence count.

COUNTEREX and fusion. COUNTEREX searches the full history for (i) competing values on the same dimension (color, sleeve length, price band), weighing behavior strength, independent observations, and context; (ii) later corrections and revoked or do-not-personalize instructions; and (iii) over-constraint, where an observed tendency is written as "only/must/never" or as invented price bounds. Its verdict is `pass`, `rewrite` (mixed or context-dependent preference, or softened strength), or `reject`. Fusion keeps an atom that passed SUPPORT on `pass`, substitutes the rewritten answer on `rewrite` while retaining provenance, and drops it on `reject`. API or parser errors are not counted as verified.

MERGE and INTENT. MERGE labels exact duplicates, semantic duplicates, and refinements of the same preference, choosing or restating a representative from the source atoms only. The existing inventory is never pruned; new items that duplicate or blend with existing ones do not count as new, incompatible atoms are reported as conflicts, and the capacity limit (20) applies to new items only. INTENT first recovers the intended query from the merged inventory without seeing Q, then compares it with Q; product substitution, audience change, lost or violated hard constraints, accessory-for-target swaps, over-narrowing, and vacuous recoveries reject the update, in which case the base inventory is kept.

WEIGHT. WEIGHT scores each atom against the user's observed post-query behavior, $w_r \in \{1, 0.5, 0, -0.5\}$: 1 when the behavior clearly supports the atom, 0.5 when the support is weak or partial, 0 when the behavior is missing, irrelevant, or inconclusive, and -0.5 when it clearly

contradicts the atom, which then joins the verifier-rejected atoms in S^-; absence of evidence is not treated as contradiction. Without post-query behavior, w_r is the support strength that SUPPORT outputs for the cited records, in $\{1, 0.5, 0.25\}$. Coverage uses $a_{x,r} = \max(w_r, 0)/\sum_{r'} \max(w_{x,r'}, 0)$.

C.11 REWARD DETAILS

Proposer reward. Each response holds up to $N_{\max}$ candidate atoms, each with its own citations, and GRPO samples four responses per prompt. The slot scores s_i of Eq. 4 are: 1 for an atom that passes every step and enters the inventory; 0.5 for one that passes SUPPORT and COUNTEREX but is removed by MERGE or by the capacity limit; -1 for a cited record missing from H (GROUND, before any model call); -0.5 for a rejection by SUPPORT or COUNTEREX, including citations that exist but do not support the claim; and 0 for an empty slot, so failing is worse than abstaining. The normalization divides by the fixed $N_{\max}$-slot weight sum, so a shorter response cannot inflate its reward. Malformed responses receive -1; infrastructure failures score zero and flag the prompt group.

Memo reward. The frozen judge, which reads no history, scores each atom in $I_k \cup \widehat{S}_k^-$ with $h_r(M) \in \{0, 0.5, 1\}$ for missing, partial, or explicit expression; $\mathrm{cov}(M)$ and $\mathrm{viol}(M)$ of Eq. 5 are the weighted mean over I_k and the plain mean over $\widehat{S}_k^-$, with $\lambda = 1$ and $\mathrm{viol} = 0$ when $\widehat{S}_k^-$ is empty. $\widehat{S}_k^-$ holds at most 10 atoms per instance; only content rejections enter it, not duplicate or capacity removals or API failures, and a rejected atom counts as expressed only when the memo asserts it as a preference of the user. Claims outside both sets are neither rewarded nor penalized. The format term is $\rho_{\mathrm{fmt}} = 0.1V + 0.1T$: V is a deterministic validity check (non-empty, prose, not JSON, 30–250 words, mentions of preferences and of the query or history, no ranking language), and T averages ten heuristic format flags with length and repetition discounts. A memo that fails the validity check receives $R_m = 0$.

C.12 TRAINING CONFIGURATION AND BUDGETS

Qwen3.5-4B is trained on four H100 GPUs and Qwen3.5-35B-A3B on 16. Training starts from the base model with memo SFT and then proposer SFT, 75 steps each, on GPT-5.4 outputs for the training set (on APeB, 1,594 memos and 1,584 proposal outputs containing 4,327 candidate atoms); memo SFT uses learning rate 5×10^{-6} and a 25,600-token sequence limit, proposer SFT 10^{-6} and 30,000 tokens, without truncation. Both routes start from this post-warm-up checkpoint. Each round of route A then runs proposer GRPO for 300 steps and memo GRPO for 200 steps; route D runs memo GRPO alone for 500 steps per round, so the two routes differ only in their GRPO phases. The SFT baseline of Table 1 is the checkpoint after memo SFT, before proposer SFT. The 4B model is trained for four rounds and the 35B-A3B model for two, with the same warm-up and per-round schedule; training would stop earlier if validation coverage stopped rising, which did not occur.

All GRPO phases use learning rate 10^{-6} in the first half of the rounds and 5×10^{-7} in the second half (rounds 1–2 and 3–4 at 4B, rounds 1 and 2 at 35B-A3B), group size 4, no KL term, the DAPO loss (Yu et al., 2025), and group reward scaling. The proposer samples at temperature 0.8 and top-p 0.95 with prompt/completion limits of 30,000/1,536 tokens; the memo role samples at temperature 1 and top-p 1 with limits 32,768/512. Prompts are drawn by cycling through the training set in a fixed order. At 4B one update consumes 16 proposer or 32 memo responses (per-device batch 1, gradient accumulation 4 or 8, i.e., 4 or 8 prompt groups), so per round A generates $300 \times 16 + 200 \times 32 = 11{,}200$ responses and D generates $500 \times 32 = 16{,}000$; at 35B-A3B both roles use 8 prompt groups per update, so A and D each generate 16,000 responses per round.

A GRPO step is one optimizer update. The matched budget of RQ2 is 500 GRPO updates per round for both routes, from the same post-warm-up checkpoint on the same inventory and data split; it excludes the shared warm-up SFT, teacher data, inventory construction, and verifier calls (Appendix C.13). Checkpoints are selected on the validation split and reported on the test split.

C.13 COMPUTE AND API COST

Table 17 reports the cost of one training round of Qwen3.5-4B on PersonaMem-v3: GPU time on four H100 GPUs, and the tokens and cost of the calls to GPT-5.6 Luna, which serves the verification

path in the proposer phase and the memo judge in the memo phase, at $0.20 per million input tokens and $1.20 per million output tokens. Token counts are estimated per phase.

Table 17: Cost of one training round on PersonaMem-v3 (Qwen3.5-4B, four H100 GPUs; API calls to GPT-5.6 Luna).

Phase	Steps	Hours	H100-hours	Input tokens	Output tokens	API cost
Memo GRPO	200	7.7	30.7	129–155M	16.0–19.2M	$44–53
Proposer GRPO	300	11–15	44–60	194–233M	24.0–28.8M	$66–80
Round	500	20–22	80–88	323–388M	40.0–48.0M	$110–133

D Extended Related Work

This appendix expands the two paragraphs of Section 5.

Reinforcement learning with verifiable feedback. Verifiable rewards enable reliable post-training when correctness can be checked by executable outcomes or deterministic evaluators (Cobbe et al., 2021; Shao et al., 2024; Guo et al., 2025); process supervision and rubric- or model-based judges extend this to open-ended generation (Lightman et al., 2024; Gunjal et al., 2026). These methods optimize a policy once the success criteria are known. In personalization the criteria themselves depend on the user, the query, and the time of the decision (Zhang et al., 2026; Liang et al., 2026). PREMO discovers what should be verified and only then trains against it.

Personalization from user histories. Language models are personalized with profiles, retrieved memories, behavioral histories, or explicit preference feedback (Salemi et al., 2024; Zhao et al., 2025; Yang et al., 2026; Rendle et al., 2009; Kang & McAuley, 2018), and recent methods optimize natural-language preference representations or update user memories from interaction feedback (Chen et al., 2025; Liu et al., 2026; Liang et al., 2026). A user-level persona is a prior that does not say which preferences are active for a given request, while a raw history shifts evidence selection onto every downstream call. We instead estimate a query-conditioned preference state, and train a memo policy to express it.

Rubric-based and self-evolving supervision. Rubrics decompose open-ended quality into interpretable criteria for scoring, data selection, and reinforcement learning (Gunjal et al., 2026), and recent systems propose or evolve rubrics from model outputs (Guan et al., 2026; Li et al., 2026; Wang et al., 2026); personalized variants derive criteria from user signals or raw histories (Zhang et al., 2026; Qiu et al., 2026). PREMO uses atomic rubric-like units as an interface, but its criteria are hypotheses about an unobserved user-specific state, so a proposal counts for nothing until it is grounded in the history, checked for counterevidence and query relevance, and weighted by the user's later behavior. Rubric evolution is thus a supervision mechanism within PREMO. The closest designs are EvoLM (Li et al., 2026), which trains one model as rubric generator and policy by alternating GRPO under a frozen judge, and EvoRubric (Guan et al., 2026), which adds layered verification and a persistent rubric pool that grows with new evaluation dimensions. Both change what the policy is rewarded for. PREMO shares the shared-model, alternating structure but studies a different learning question: whether training the model to discover evidence-supported preferences improves the same model's later memo, over and above the verified atoms that discovery adds to the inventory. The proposer therefore does not verify its own claims, the memo policy never reads the inventory, and Section 4.3 and Appendix C.8 separate the checkpoint effect from the supervision effect.

E Prompts

This appendix gives condensed versions of the prompts used in training, in the order of Figure 2, followed by the memo judge and the PersonaMem-v3 task judge; the full prompts are released with the code. They are shown in their PersonaMem-v3 form; the APeB form differs in the history fields and adds the Weight step.

Proposer (condensed)

```
Propose up to N NEW query-relevant preference atoms.

Input:
- current query
- indexed prior conversation and observable history
- existing atoms in full

For each candidate, produce one atomic question, answer it only from
the supplied history, explain why it is missing, and cite 1-3 real
source/source_index records. Do not repeat existing atoms, infer a
preference from the query alone, or invent history. If support is
unreliable, abstain with has_answer=false; zero candidates is allowed.

Return JSON only:
{
  "positive_candidates": [
    {
      "candidate_label": "CANDIDATE_1",
      "candidate_type":
"query_relevant_preference|avoidance_preference",
      "aspect": "...",
      "question": "...",
      "why_missing": "...",
      "has_answer": true,
      "answer": "...",
      "history_evidence": [
        {"source": "prior_conversation|observable_history",
         "source_index": 0, "support_reason": "..."}
      ]
    }
  ],
  "negative_candidates": [],
  "short_explanation": "..."
}
```

Support (condensed)

```
Verify one filled preference atom.

Input:
- current query
- one flat filled atom
- only the deterministically resolved cited history records

Judge independently:
1. history_supported: cited user evidence supports the answer and
polarity;
2. query_relevant: the preference can materially affect a plausible
answer;
3. privacy_safe: the atom does not use revoked / forbidden information;
4. support_strength in {1, 0.5, 0.25}: how strongly the cited behavior
supports the answer (1: direct support, e.g. a purchase or repeated
behavior; 0.5: partial support; 0.25: weak support, e.g. a single
view).

Do not search uncited history, repair the answer, or reward keyword
overlap.
Return JSON only with the three booleans, support_strength, accept,
short reasons, and reject_reason.
```

Counterex (condensed)

```
Review one anchor-accepted atom against the complete bounded visible
context.

Check only:
- strong same-aspect alternatives or contradictions;
- later corrections or context-specific updates;
- revocation / do-not-personalize instructions;
- unsupported exclusivity, permanence, or over-narrowing.

Do not redo positive support discovery or general query-fit scoring.
Return one decision: pass, rewrite, or reject. Rewrite may change only
the answer and must preserve the supported core; reject when the claim
is revoked, counter-dominated, or cannot be safely repaired.

Output JSON only:
{
  "decision": "pass|rewrite|reject",
  "issue_type": "none | mixed_same_aspect | later_correction |
revoked_information | overrestriction | counter_dominant",
  "reason": "...",
  "evidence_references": [{"source": "...", "source_index": 0}],
  "rewritten_answer": null
}
```

Merge (condensed)

```
Given the flat verified atom set, keep every distinct actionable
preference.

Remove exact duplicates and directional semantic duplicates when one
retained atom preserves all material meaning, scope, qualifiers, and
exceptions of another. Report genuinely incompatible atoms as conflict
groups. Do not rewrite, merge text, rank preferences, re-check history
support, or resolve a conflict.

Return JSON only:
{
  "kept_atom_indexes": ["R1"],
  "duplicate_groups": [
    {"kept": "R1", "duplicates": ["R2"],
     "relationship": "exact_duplicate|semantic_duplicate", "reason":
"..."}
  ],
  "conflict_groups": [
    {"atom_indexes": ["R3", "R4"], "reason": "..."}
  ],
  "short_explanation": "..."
}
```

Intent, step 1: query recovery (condensed)

```
Infer the task scope and response intent represented by the atom set.

Use only atom_index, candidate_type, aspect, question, and answer. Do
not assume the hidden query, history, recipient, domain, or product
category. Keep positive preferences separate from avoidances. If scope
is not recoverable, return "unknown task scope" and honest ambiguity
instead of inventing one.

Return JSON only:
{
  "recovered_task_scope": "...",
  "recovered_intent": "...",
```

```
  "positive_preferences": ["..."],
  "avoidance_preferences": ["..."],
  "ambiguity": "low|medium|high",
  "reason": "..."
}
```

Intent, step 2: gate (condensed)

```
Decide whether recovered_task_scope and recovered_intent can usefully
personalize the original user query without changing its task, topic,
recipient/context, or explicit hard constraints.

Do not inspect individual atoms, history, evidence, preference lists,
or verifier results. Reject only major set-level drift, incoherence,
or content-free recovery; do not reject merely for incomplete coverage
or valid specialization of a broad query.

Return JSON only:
{
  "accept": true,
  "drift_type": "none | wrong_task | wrong_topic |
wrong_recipient_or_context | lost_explicit_constraint |
unsupported_over_narrowing | generic_empty | incoherent_set | unknown",
  "reason": "..."
}
```

Weight, APeB (condensed)

```
Decide whether each atom is consistent with the user's post-query
behavior.

Assign weight 1 when the behavior clearly supports the atom, 0.5 when
the support is weak or partial, 0 when the behavior is missing,
irrelevant, or inconclusive, and -0.5 when the behavior clearly
contradicts the atom.

Use only observed post-query behavior. Do not infer missing
preferences, treat absence of evidence as contradiction, or reconsider
how the atom was recovered.

Return JSON only:
{
  "relation": "match|weak_match|uncertain|violation",
  "weight": "1|0.5|0|-0.5",
  "reason": "..."
}
```

Memo policy, PersonaMem-v3 (condensed)

```
You are a preference-memory writer for a personalized assistant.

Input:
- current_query
- compacted_pre_query_context

Recover only history-supported preferences or avoidances that can
materially improve this query. Lead with one specific, high-confidence
primary preference; preserve scope, recipient, uncertainty, later
corrections, and privacy boundaries.

Do not answer the query, select an item, expose history the user has
marked as private or not to be personalized, invent a preference, or
reuse revoked / do-not-personalize information.
```

```
Output plain prose only. If nothing reliable remains, return the fixed
no-additional-preference fallback.
```

Memo judge, coverage (condensed)

```
Input: the query, the supported atom set E(x) = {(e_i, w_i)} with
importance weights w_i > 0, and the memo.

For each supported atom e_i, judge whether the memo states the
corresponding preference. Assign 1 for a clear and correct match, 0.5
for a partial or implicit match, and 0 for a missing, generic,
off-query, or contradicted match. Do not give credit for preferences
outside the supplied atom set. Use the provided weights when
aggregating the scores.

Coverage = sum_i w_i s_i / sum_i w_i, with s_i in {0, 0.5, 1}.
```

Memo judge, faithfulness (condensed)

```
Input: the query, the visible user history, and the memo.

Score from 0 to 1 how much the memo makes unsupported or overconfident
preference claims. A score of 0 means its claims are supported by the
query or visible history and appropriately calibrated; a score of 1
means the memo contains severe or pervasive unsupported claims. Use
intermediate values for partial violations.

Faithfulness = 1 - this violation score.
```

Memo judge, grounding (condensed)

```
Input: the query, the visible user history with identifiable records,
and the memo.

1. Split the memo into sentence-level units.
2. Decompose each sentence into minimal, independently verifiable
atomic claims.
3. For every atomic claim, identify the history record or records that
the claim explicitly cites or implicitly relies on.
4. Verify that each referenced record exists in the visible history
and that its content supports the claim.
5. Label every atomic claim as supported (the records exist and fully
support the claim), partial (the records exist but support the claim
only partially, indirectly, or with weaker confidence than the memo
expresses), or unsupported (the records do not exist, contradict the
claim, or provide no support).

All atomic claims enter the denominator. Grounding = (N_supported +
0.5 N_partial) / (N_supported + N_partial + N_unsupported).
```

PersonaMem-v3 judge, task completion (condensed)

```
Score task_completion from 0 to 10.

Judge correctness, feasibility, factual reliability and useful
completion of the actual request. Penalize concrete functional
failures: an explicit request left unfulfilled, an incorrect external
fact, a fabricated or unfindable recommendation, or execution advice
that does not work.
```

```
Judge this axis independently of personalization. A generic but
correct and useful answer may score highly. Do not mechanically
subtract another fixed penalty merely because a preference or rule
issue exists. Lower completion for a rule violation only when it
creates an actual, separately explained loss of task functionality.
```

PersonaMem-v3 judge, preference alignment (condensed)

```
Score preference_alignment from 0 to 10.

Credit only positive, supported and permitted personalization that
improves this particular answer. Evaluate the route actually chosen;
do not require every preference or the most frequent alternative.

A useful generic answer is normally around 5-6. Scores 9-10 require
specific, well-supported choices that materially improve the response
to this query.

Give no positive credit to prohibited, fabricated, overstated or
context-mismatched personalization. Do not add a second numerical
punishment for a rule violation here; score the remaining permitted
personalized content, while rule penalties belong on rule_compliance.
```

PersonaMem-v3 judge, rule compliance (condensed)

```
Score rule_compliance from 0 to 10.

Penalize only demonstrated violations of an applicable explicit
boundary or hard constraint, fabricated personal facts, or definite
misuse of a restricted historical basis. Do not treat an omitted
optional preference, awkward style, ordinary factual error, or
uncertain scope as a rule violation.

If no applicable violation is established, score 10. Unknown or
ambiguous scope alone must not reduce the score. The worst supported
issue and its actual extent determine the severity band; do not score
by counting unrelated rules or citations.
```