QUIET: QA-FREE UNBIASED IMPLICIT PREFERENCE EXTRACTION WITH SEMANTIC F1 REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

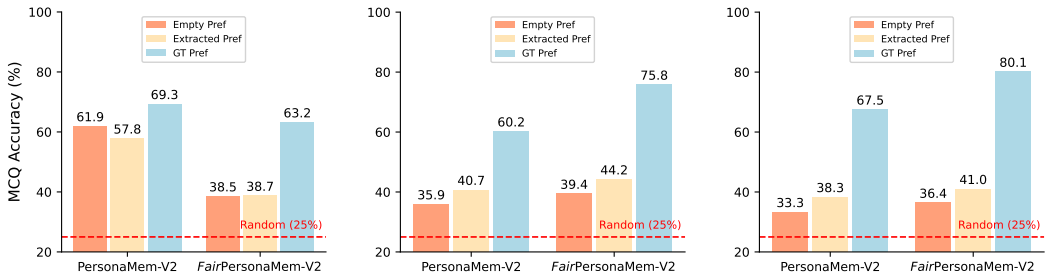
Personal LLM agents rely on preferences extracted from user–agent interactions to provide personalized responses. However, such preferences are often expressed only implicitly across lengthy interactions, making reliable extraction challenging. Although specialized small-scale extractors offer an efficient solution, we observed that question-answering (QA) rewards used in reinforcement learning (RL) can bias extractors toward QA-specific patterns at the expense of preference fidelity. Specifically, even on the rigorous PersonaMem-V2 benchmark, the QA-reward-trained model performs worse when conditioned on its own extracted preferences than without them, highlighting the need for improved extraction methods and new benchmarks that do not overlook such biases. To this end, we introduce *Fair*PersonaMem-V2, a bias-aware reconstruction of the benchmark, and QUIET (**Q**A-free **U**nbiased **I**mplicit preference **E**x**T**raction), a reward design for RL to train a preference extractor directly from ground-truth preferences, without requiring QA supervision. On *Fair*PersonaMem-V2, the small-scale Qwen3-4B-Instruct extractor trained with QUIET achieves a 32% relative MCQ accuracy gain and an $11\times$ higher semantic F1, with $21\times$ fewer training context tokens compared to the QA-reward trained baseline.

1 INTRODUCTION

Large language models (LLMs) are increasingly being deployed as personal agents that maintain long-term relationships with individual users. Their usefulness depends on accurately capturing and recalling user preferences. However, users rarely express these preferences explicitly; instead, they reveal them implicitly over the course of multi-turn conversations. A capable agent should therefore extract implicit preferences from conversations and leverage them to provide personalized responses (Nie et al., 2026). Growing interest in this problem has produced several benchmarks (Zhao et al., 2025; Jiang et al., 2025a) that measure preference extraction ability from long conversations through downstream question answering. Approaches to this problem have gradually shifted from prompt-based pipelining (Gao et al., 2024; Zhao et al., 2026) toward training dedicated extractors with conventional QA-derived RL rewards (Wu et al., 2025; Yan et al., 2026).

A common evaluation metric in this field is QA accuracy (Ma et al., 2026a; Salemi & Zamani, 2025). However, it is not a suitable RL reward for training preference extraction because it also reflects a model's answering ability. In QA-reward RL training, preferences extracted from a conversation are used to answer a downstream question, and answer correctness is assigned as a reward to the entire extract-and-answer trajectory (Chen et al., 2025; Yu et al., 2026). Consequently, the model is rewarded not only for extracting the correct preference but also for selecting the correct answer under a particular QA distribution. This creates an opportunity for **QA shortcutting**, a form of reward hacking: the model can obtain high rewards by learning recurring query–choice patterns during answer generation, even when its extraction does not improve. Because existing benchmarks, such as PersonaMem-V2 (Jiang et al., 2025b), retain these patterns in their questions and choices, they may overestimate a model's preference-extraction ability. As shown in Figure 1a, a baseline trained with the QA-reward in the MemAgent framework (Yu et al., 2026) achieves 61.9% multiple-choice question (MCQ) accuracy without any preference information—well above the 25% chance level for four-option QA—and even outperforms the same model when it uses its own extracted preferences (57.8%). This result provides clear evidence that the model has learned a query–choice shortcut rather than learning to extract preferences more accurately.

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.



(a) Qwen3-4B **RL-trained** Answerer (b) Qwen3-32B Answerer (c) GPT-5.4-mini Answerer

Figure 1: **QA-reward RL can exploit query–choice shortcuts.** The QA-reward GRPO (Shao et al., 2024) trained Qwen3-4B-Instruct answerer achieves 61.9% MCQ accuracy with empty preferences, exceeding 57.8% accuracy obtained with its own extracted preferences on PersonaMem-V2. On proposed *Fair*PersonaMem-V2, where semantic patterns between queries and choices have been removed, the accuracy drops sharply to 38.7% (a). In contrast, off-the-shelf answerers, Qwen3-32B and GPT-5.4-mini, exhibit similar performance trends across both benchmarks (b, c).

To evaluate preference extraction independently of query-choice shortcuts, we first construct *Fair*PersonaMem-V2 by rebuilding PersonaMem-V2 with choice-neutral queries and semantically plausible distractors, making the relevant preference essential for distinguishing the correct answer as in Figure 1. Off-the-shelf models remain stable across both benchmarks, suggesting that *Fair*PersonaMem-V2 preserves general QA capability. In contrast, the QA-reward RL-trained baseline drops sharply from 57.8% to 38.7% with its own extracted preference provided, revealing its reliance on the specific QA distribution. We describe *Fair*PersonaMem-V2 in detail in Section 4.1 and use it as the main benchmark to evaluate the generalizable ability of the model.

Then, to disentangle preference extraction from question answering during model training, we propose QUIET (**Q**A-free **U**nbiased **I**mplicit preference **Ex**T**r**action), a reward design for evaluating preference extraction quality without question answering. By eliminating QA from reward computation, QUIET enables RL training that is unbiased by any QA distribution. We first formulate preferences as a set of semantic-unit bullets, and then define an F1 score calculated by semantic bipartite matching between the extracted bullets and ground-truth (GT) bullets. This semantic F1 score serves as a reward that evaluates the quality of extraction for a persona's overall preferences. Since F1 score is a harmonic mean of precision and recall, semantic F1 score alone can penalize both preference omissions and inaccurate additions. Moreover, QUIET decouples preference extraction from specific QA configurations and allows flexible selection of the answer model at inference time.

QUIET consistently generalizes better than QA-reward approaches in decorrelated and cross-benchmark evaluations. We conduct controlled comparisons within the MemAgent framework to identify the causes of shortcut learning. We compare the approaches using Qwen3-4B-Instruct, Qwen3-1.7B (Yang et al., 2025), and Llama3.2-3B (AI, 2024) as extractor backbones. We evaluate them on PersonaMem-V2, *Fair*PersonaMem-V2, as well as in cross-benchmark evaluations: PersonaMem (Jiang et al., 2025a) and AlpsBench (Xiao et al., 2026). QUIET yields consistent generalization gains in QA accuracy across all backbones even without using QA data during training. With the 4B backbone, QUIET achieves 47.7% MCQ accuracy with free-text generation and 49.8% with an edit-based output format on *Fair*PersonaMem-V2 with a 32k input context length, outperforming QA-reward-based joint RL, which achieves 37.6%. QUIET achieves these gains while reducing the number of training context tokens per epoch by $21\times$, from 747M to 36M.

Contributions.

- We identify QA-reward answerer bias in preference extraction, where models exploit recurring query-choice patterns instead of improving their extraction ability. To isolate this behavior, we construct FairPersonaMem-V2, a controlled, decorrelated variant of PersonaMem-V2 with choice-neutral queries and semantically plausible distractors.
- We propose QUIET, a QA-free RL reward for implicit preference extraction. QUIET directly evaluates extracted preference sets using a semantic bipartite-matching F1 score, thereby decoupling extraction training from specific QA distributions and answer models.

- Through controlled experiments across multiple extractor backbones and benchmarks, we show that QUIET consistently generalizes better than QA-reward approaches under decor-related and cross-benchmark evaluations.

2 Related Work

2.1 Personalized Memory Systems

Memory Agents. Personalized memory systems for LLM-based agents initially relied on prompt-based organization, heuristic retrieval, and explicit context management. Generative Agents (Park et al., 2023) combined retrieval and reflection over a memory stream, MemGPT (Packer et al., 2023) managed context through OS-style paging, and Mem0 and A-MEM (Chhikara et al., 2025; Xu et al., 2025) used vector retrieval and structured note-taking. Later approaches introduced compact, structured representations: AtomMem (Yao et al., 2026) organizes atomic facts in an associative graph, while EverMemos (Hu et al., 2026) consolidates episodic traces into semantic structures through a three-stage lifecycle. PersonaAgent (Zhang et al., 2026c) connects memory to action via per-user persona prompts, and LightMem (Zhang et al., 2026a) delegates online retrieval and offline consolidation to a smaller language model. Moving beyond prompt- and memory-engineering approaches, RL methods learn memory-management policies directly. Memory-R1 (Yan et al., 2026) jointly trains a memory manager and answer agent with outcome rewards, MEM1 (Zhou et al., 2026) learns to maintain a constant-size state through pruning, and Mem-α (Wang et al., 2025) combines QA accuracy with multiple shaping rewards. More recent methods supervise memory updates directly: DeltaMem (Zhang et al., 2026b) rewards memory-state transitions using Memory-based Levenshtein Distance, MMPO (Liu et al., 2026) penalizes uncertain updates through Belief Entropy, and EvoArena (Xu et al., 2026b) evaluates memory evolution in dynamic environments.

2.2 Preference Extraction

Preference Awareness Benchmarks. Learning to extract implicit user preferences from conversations has emerged as a distinct problem within personalized memory. PersonaMem (Jiang et al., 2025a) established the benchmark for this task, evaluating extraction quality through multiple-choice questions over long dialogue histories, and PersonaMem-V2 (Jiang et al., 2025b) scaled it to 1,000 personas for both training and evaluation. Additional benchmarks extend this scope: PrefEval (Zhao et al., 2025) measures preference following across 3,000 pairs spanning both explicit and implicit forms, AlpsBench (Xiao et al., 2026) targets preference alignment in real dialogues, and Horizon-Bench (Li et al., 2026) focuses on evolving preferences over time.

Preference Extraction Approaches. Several methods train preference extraction policies with reinforcement learning. MemAgent (Yu et al., 2026), the RL baseline introduced for PersonaMem-V2, recurrently updates a compact preference memory and optimizes it with downstream QA accuracy as the reward. Inside Out (Zhao et al., 2026) organizes preferences into a schema-constrained PersonaTree and trains tree operations with process-reward RL. MemCoE (Xu et al., 2026a) first induces natural language memory guidelines through contrastive trajectory feedback, then aligns a memory policy against those guidelines using process rewards. Weak rewards from RAG interactions have also been used to train preference models (Hao et al., 2026), while synthetic interaction data can support scalable personalization training (Ma et al., 2026b). Although these methods use diverse reward sources, downstream QA outcomes remain a common training signal for preference extraction. Our work further diagnoses that QA rewards can induce benchmark-specific shortcut behavior and introduces QUIET, an approach that is completely independent of downstream QA tasks.

3 Method

3.1 Problem Formulation

We study implicit preference extraction from long user-agent conversations. Let $\mathcal{H}$ denote an ordered conversation history partitioned into fixed-length chunks $(c_1, \ldots, c_K)$, each of length C. Starting from an empty preference state $\hat{\mathcal{P}}_0$, a preference extraction model f sequentially updates the prefer-

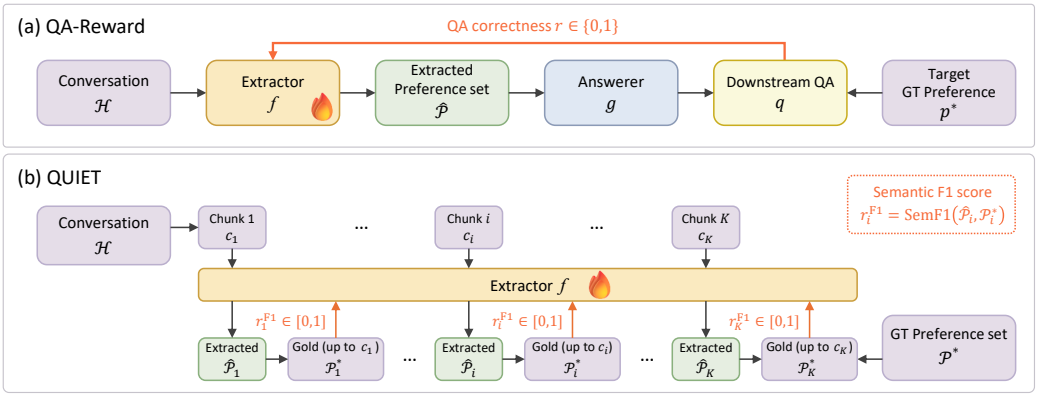


Figure 2: **Overview of QUIET method in comparison with conventional QA-reward RL**

ence state as

$$\hat{\mathcal{P}}_i = f(c_i, \hat{\mathcal{P}}_{i-1}), \qquad i = 1, \ldots, K, \tag{1}$$

subject to a fixed token budget $|\hat{\mathcal{P}}_i| \leq L$. At each step, $\hat{\mathcal{P}}_i$ should be complete and standalone, containing all personalization-relevant preferences implied by the conversation chunks $(c_1, \ldots, c_i)$ that are necessary to answer subsequent queries without access to the original chunks. The extractor retains only the current preference state and does not revisit earlier chunks. The final state $\hat{\mathcal{P}} = \hat{\mathcal{P}}_K$ represents the preferences extracted from the entire conversation.

Given a subsequent query q, an answer model g produces an answer

$$\hat{y} = g(\hat{\mathcal{P}}, q). \tag{2}$$

Preference extraction is query-independent because $\hat{\mathcal{P}}$ is constructed before q is provided. Unlike RAG-based memory agents that retrieve query-specific context at response time, our extractor constructs a query-independent preference state in advance. In our MCQ evaluation, $\hat{y}$ denotes the final answer choice selected by the answer model.

3.2 QUIET: QA-free Unbiased Implicit preference ExTraction

The sequential formulation above specifies what the extractor must retain after each conversation chunk, but does not specify how it should be trained. Conventional QA-reward training evaluates only the final preference state through downstream QA correctness:

$$r^{\text{QA}} = \mathbb{I}\left[g(\hat{\mathcal{P}}_K, q) = y^\star\right]. \tag{3}$$

This reward depends jointly on the extracted state, the answer model, and the query and answer-choice distribution. Consequently, the same reward can increase because the extraction improves or because the answerer exploits regularities in the QA task.

We propose QUIET to remove this ambiguity by supervising the preference state itself. For each chunk $(c_1, \ldots, c_i)$, let $\mathcal{P}_i^\star$ denote the gold preference state implied by that chunk. It contains exactly the GT preferences associated with chunks $(c_1, \ldots, c_i)$ and excludes preferences that appear in the later chunks. After the update $\hat{\mathcal{P}}_i = f_\theta(c_i, \hat{\mathcal{P}}_{i-1})$, QUIET assigns the step-level reward computed using semantic F1 score (SemF1),

$$r_i^{\text{F1}} = \text{SemF1}\left(\hat{\mathcal{P}}_i, \mathcal{P}_i^\star\right). \tag{4}$$

Since neither a query q, a target answer y, nor an answerer g participates in this computation, training remains entirely unbiased with respect to the QA task. We assign a chunk-specific target: comparing an early state extraction against the final gold state would penalize the extractor for preferences that have not yet appeared. Equation 4 instead provides a local learning signal after every chunk, yielding K extraction rewards for a K-chunk conversation rather than a single terminal QA reward. The overview of conventional QA-reward RL and our QUIET is demonstrated in Figure 2.

3.3 SEMANTIC F1 SCORE FOR PREFERENCES

Preference states admit many semantically equivalent textual realizations, making exact string matching unsuitable as an extraction reward. We therefore adopt semantic dimension matching, beginning by decomposing each state into a set of atomic preference bullets. Let

$$\hat{\mathcal{B}}_i = \{\hat{p}_{i,1}, \ldots, \hat{p}_{i,m_i}\}, \qquad \mathcal{B}_i^\star = \{p_{i,1}^\star, \ldots, p_{i,n_i}^\star\} \tag{5}$$

denote the bulletized representations corresponding to $\hat{\mathcal{P}}_i$ and $\mathcal{P}_i^\star$, respectively. The preference state should be represented as semantic bullets before scoring.

An LLM judge then performs bipartite matching between the extracted and gold bullet sets. For each extracted bullet $\hat{p}_{i,j}$, the resulting match identifies either a corresponding gold bullet or no match,

$$z_{i,j} \in \{0, 1, \ldots, n_i\}. \tag{6}$$

Here $z_{i,j} = k$ indicates that $b_{i,j}$ preserves the essential meaning and distinguishing attributes of $p_{i,k}^\star$, and $z_{i,j} = 0$ indicates no valid match. Each extracted bullet may select at most one gold preference. We count the number of uniquely covered gold preferences as

$$a_{i,k} = \mathbb{I}\left[\exists j \in \{1, \ldots, m_i\} \text{ s.t. } z_{i,j} = k\right], \qquad t_i = \sum_{k=1}^{n_i} a_{i,k}, \tag{7}$$

where $a_{i,k}$ indicates whether the k-th gold preference is matched by at least one extracted bullet.

For non-empty $\hat{\mathcal{B}}_i$ and $\mathcal{B}_i^\star$, semantic precision and recall are t_i/m_i and t_i/n_i, respectively, and their harmonic mean, the F1 score, reduces to $2t_i/(m_i + n_i)$. We define the complete score as

$$\mathrm{SemF1}\left(\hat{\mathcal{P}}_i, \mathcal{P}_i^\star\right) = \mathrm{SemF1}(\hat{\mathcal{B}}_i, \mathcal{B}_i^\star) = \begin{cases} 1, & m_i = n_i = 0, \\ 0, & m_i = 0 \text{ or } n_i = 0, \\ \dfrac{2t_i}{m_i + n_i}, & \text{otherwise.} \end{cases} \tag{8}$$

Unmatched or duplicated extractions lower precision, whereas missing gold preferences lower recall. By balancing these two aspects, the F1 reward jointly promotes accurate and comprehensive preference extraction. Moreover, $r^{F1} \in [0, 1]$ provides a continuous learning signal, allowing partial improvements in the extracted memory to be immediately reflected during training. In contrast, the binary QA reward $r^{QA} \in \{0, 1\}$ remains unchanged unless the final answer flips from incorrect to correct, even when the quality of the extracted memory improves.

3.4 PREFERENCE-STATE UPDATES

QUIET is agnostic to how the transition $f_\theta(c_i, \hat{\mathcal{P}}_{i-1})$ is serialized. We instantiate it with two complementary output formats, Plain and Diff.

Plain. The model directly generates the complete next state as plain-text,

$$\hat{\mathcal{P}}_i \sim f_\theta(\cdot \mid c_i, \hat{\mathcal{P}}_{i-1}), \tag{9}$$

This format makes no assumptions about explicit edit operations, but requires the model to reproduce every preference that should persist whenever the state is updated.

Diff. The model instead generates an edit Δ_i composed of Mem0-style Chhikara et al. (2025) `ADD`, `UPDATE`, and `DELETE` operations over key–value entries. Details of the operations and entries are provided in Section A.3. A deterministic update operator U applies the edit to the previous state:

$$\Delta_i \sim f_\theta(\cdot \mid c_i, \hat{\mathcal{P}}_{i-1}), \qquad \hat{\mathcal{P}}_i = U(\hat{\mathcal{P}}_{i-1}, \Delta_i). \tag{10}$$

Because Diff model generates only the change, its output length does not grow with the complete accumulated state since remaining entries need not be regenerated. We additionally define a binary format reward for Diff that indicates whether the generated edit conforms to the edit schema:

$$r_i^{\mathrm{fmt}} = \mathbb{I}[\Delta_i \text{ is valid under the edit schema}]. \tag{11}$$

During training, the normalized format reward is weighted by λ_{fmt} and combined with the normalized semantic F1 reward. Further details on reward normalization and aggregation are provided in Section A.4. Despite differing in their output formats, both Plain and Diff extractors produce a bounded and standalone preference state as defined in Section 3.1.

4 EXPERIMENTS

4.1 *Fair*PERSONAMEM-V2 BENCHMARK

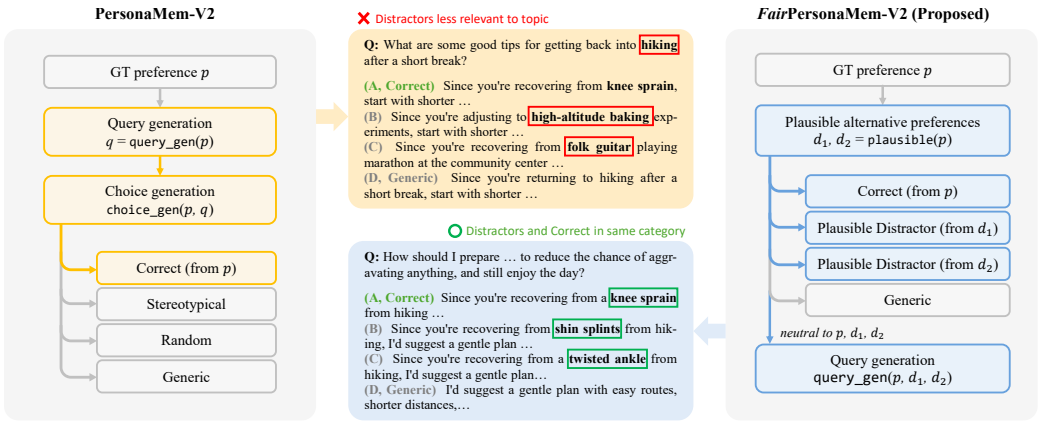


Figure 3: **QA construction procedures and representative comparison in PersonaMem-V2 and *Fair*PersonaMem-V2.** In PersonaMem-V2, the query is generated first from a single GT preference p, then the correct choice is derived from (p, q); both are strongly aligned with p, making the correct choice identifiable without the extracted preference. In *Fair*PersonaMem-V2, plausible distractors d_1, d_2 that belong to the same category as p are generated, and the query is generated from all three preferences (p, d_1, d_2), so no surface cue in the query points to the correct choice.

PersonaMem-V2 Jiang et al. (2025b) is a benchmark that evaluates user preferences extracted from multi-turn long conversation using four-option multiple-choice questions (MCQs). Each persona in the dataset is assigned to GT preference set $\mathcal{P} = \{p_1, \ldots, p_n\}$. To verify how well an extractor has captured the target preferences, each MCQ item is constructed for a preference $p \in \mathcal{P}$, and an answer model selects the correct choice referring to the extracted preference set $\hat{\mathcal{P}}$. The central weakness of PersonaMem-V2 is that its QA construction can reveal the correct choice without any preference information. The root cause is query–choice bias: each QA item is generated from a single preference $p \in \mathcal{P}$, where the query follows $q = \texttt{query_gen}(p)$ and the correct choice follows $c = \texttt{choice_gen}(p, q)$, so both are strongly aligned with p. Here $\texttt{query_gen}$ and $\texttt{choice_gen}$ denote LLM-based dataset generation to produce queries and choices, respectively.

We propose *Fair*PersonaMem-V2, a reconstruction of PersonaMem-V2 that replaces its queries and choices removing the query-choice bias, while retaining the original chat histories and GT preferences. Retaining these elements preserves the same personas, long-context evidence, and extraction targets, so any performance difference can be attributed to QA reconstruction rather than to a changed preference extraction task. The reconstruction proceeds in two steps. First, replace less relevant distractors to plausible distractors reflecting plausible alternative preferences (d_1, d_2). Then, generate decorrelated query neutral to correct choice and distractors. The plausible distractors serve as hard negatives to the given query that shares the topic and response structure of the correct choice, so an answerer cannot reject them simply without referring the proper preference. The overall differences between the two benchmark construction procedures are illustrated in Figure 3. We used GPT-5.4 OpenAI (2026) for *Fair*PersonaMem-V2 generation.

4.2 IMPLEMENTATION DETAILS

We use various language models: Qwen3-4B-Instruct, Qwen3-1.7B (Yang et al., 2025), and Llama3.2-3B (AI, 2024) for experiments. The vanilla models are used for the retrieval-based methods, whereas the trained models are used for the complete-form extraction methods. For training, we first perform supervised fine-tuning (SFT), followed by GRPO training, both on the PersonaMem-V2 training set. SFT is performed for 400 steps with a learning rate 10^{-5} and batch size of 64. The models are then trained using GRPO for one epoch with a learning rate 10^{-6}, 20 warmup steps, batch size of 16, and $G = 8$ rollouts per input. A format reward weight $w_{\text{fmt}} = 0.2$ is used for QUIET-diff. Conversations are divided into chunks of 5k tokens and processed sequentially, with extraction

Table 1: **Progressive diagnostic comparison.** The consecutive comparisons isolate the bias of a jointly RL-trained answerer (QA-RL-Joint vs. QA-RL-Swap), the effect of extractor updates under QA reward (QA-RL-Swap vs. QA-RL-Frozen), and the benefit of semantic F1 reward (QA-RL-Frozen vs. QUIET). All experiments are conducted with Qwen3-4B-Instruct as the extractor backbone, on 32k input context length. We report MCQ accuracy (%) on PMv2 and *Fair*PMv2, together with semantic F1 score (%) measured on their shared extraction.

Configuration	Answerer (train → eval)	Reward	PMv2	*Fair*PMv2	Semantic F1
QA-RL-Joint	Qwen3-4B → Qwen3-4B	QA correctness	**57.8**	37.6	4.4
QA-RL-Swap	Qwen3-4B → GPT-5.4-mini	QA correctness	38.3	41.0	4.4
QA-RL-Frozen	GPT-5.4-mini → GPT-5.4-mini	QA correctness	43.6	44.0	3.2
QUIET-plain	✗ → GPT-5.4-mini	Semantic F1	47.9	<u>47.7</u>	<u>40.7</u>
QUIET-diff	✗ → GPT-5.4-mini	Semantic F1 + Format	<u>51.1</u>	**49.8**	**47.5**

budget bounded by $L = 2,048$ tokens. Thinking mode is enabled for the Qwen3 models during RL training and inference. Unless otherwise specified, GPT-5.4-mini is used as the semantic-matching judge for F1 score calculation and as the MCQ answerer, while Sonnet-4.6 is used as plain-text bulletizer and LLM judge of AlpsBench. Additional implementation details are provided in Section A.1. The methods are evaluated on PersonaMem-V2 (Jiang et al., 2025b) (PMv2), *Fair*PersonaMem-V2 (*Fair*PMv2), PersonaMem (Jiang et al., 2025a) (PMv1), and AlpsBench (Xiao et al., 2026). For AlpsBench, four evaluation dimensions of Task 4: Preference Following (PF), Virtual-Reality Awareness (VRA), Constraint Following (CF), and Emotional Intelligence (EI) are used.

4.3 Results

Progressive comparison. Table 1 diagnoses why QA-reward RL training does not necessarily yield a generalizable extractor. QA-RL-Joint and QA-RL-Swap use the same jointly trained extractor but different answerers at evaluation. Replacing the jointly trained answerer with an external answerer substantially reduces accuracy on PMv2 while improving performance on the decorrelated benchmark, indicating that extractor–answerer co-adaptation biases the jointly trained answerer. By training only the extractor against a frozen answerer, QA-RL-Frozen improves accuracy on both benchmarks compared to QA-RL-Swap, mitigating this answerer bias. However, its persistently low semantic F1 suggests that QA-reward remains an insufficient objective for learning faithful extractions. In contrast, QUIET directly rewards extraction results without any QA process during training, yet achieves higher QA accuracy on both benchmarks. This demonstrates that directly optimizing extraction quality is more effective than indirect QA-reward RL and generalizes better to downstream QA. Formal definitions of comparison configurations are provided in Section A.5.

Main results. Table 2 shows that QUIET achieves strong performance on decorrelated and out-of-distribution (OOD) benchmarks across extractor sizes and context lengths. In most settings, QUIET-diff is the best-performing complete-form extractor, while QUIET-plain ranks second, although their relative ordering varies in some settings. With the 4B extractor, QUIET-diff reaches 49.8% on *Fair*PMv2 32k and 64.2% on PMv1 32k, surpassing QA-RL-Joint by 12.2%p and 7.9%p respectively. This pattern holds in most settings for Llama3.2-3B and Qwen3-1.7B extractor backbones. A different pattern occurs on the in-distribution (ID) PMv2 benchmark, where QA-RL-Joint achieves exceptionally high performance. As diagnosed in Table 1, however, this improvement stems from the biased, jointly trained answerer and does not translate into consistent gains beyond the ID benchmark. On AlpsBench, QUIET approaches also achieve the strongest results across the evaluation dimensions. It shows that the benefit of directly optimizing extraction quality extends beyond question answering to open-ended preference-conditioned generation.

The performance gap between QUIET-diff and other methods, including QUIET-plain, widens consistently as context length increases, showing that the diff format is particularly effective for long-context settings. In PMv1 with a 4B extractor, the gap increases from 2.9%p at 32k to 12.2%p at 128k and 18.9%p at 1M. The same trend also holds for smaller extractors. Other methods must regenerate the full cumulative preferences from scratch at every chunk, so errors in preserving existing content accumulate as context grows. Diff mitigates this by outputting only edit operations, reducing the burden on small models when handling long contexts.

Table 2: **Comparison of results across extraction models.** PersonaMem-V2, *Fair*PersonaMem-V2, and PersonaMem report MCQ accuracy (%). Semantic F1 reports the average semantic F1 score (%) on shared extraction of PMv2 and *Fair*PMv2 against the GT preferences. AlpsBench reports the average based on binary ratings for PF, VRA, and CF, and on a 1–5 rating scale for EI. Within Complete-form Extraction, **bold** marks the best and underline marks the second best per column.

Model	Method	ID		Decorrelated				OOD						
		PMv2		*Fair*PMv2		Semantic F1		PMv1			AlpsBench			
		32k	128k	32k	128k	32k	128k	32k	128k	1M	PF	VRA	CF	EI
Qwen3-4B-Instruct	*Retrieval-based*													
	Mem0	39.9	39.2	39.9	39.2	–	–	60.3	55.3	46.9	0.485	0.383	0.549	3.39
	A-MEM	41.3	40.8	41.3	40.8	–	–	49.1	48.3	47.2	0.311	0.392	0.563	2.93
	LangMem	39.8	38.0	39.8	38.0	–	–	49.9	44.9	41.1	0.316	0.401	0.563	3.05
	Complete-form Extraction													
	SFT	33.9	35.7	26.1	28.3	4.8	4.2	50.3	43.1	34.4	0.447	0.460	0.396	2.43
	QA-RL-Joint	**57.8**	**61.7**	37.6	38.3	4.4	3.3	56.5	42.9	42.0	0.436	0.481	0.421	2.34
	QA-RL-Swap	38.3	35.3	41.0	38.2	4.4	3.3	56.4	42.8	34.7	0.590	0.535	0.596	2.85
	QA-RL-Frozen	43.6	35.0	44.0	40.5	3.2	0.7	33.3	29.2	27.7	0.594	0.562	0.600	2.74
	QUIET-plain	47.9	41.3	47.7	40.8	40.7	6.5	61.3	46.1	34.7	0.592	0.570	0.587	**2.91**
	QUIET-diff	51.1	51.8	**49.8**	**44.2**	**47.5**	**27.1**	**64.2**	**58.3**	**53.6**	**0.598**	**0.587**	**0.653**	2.89
Llama3.2-3B	*Retrieval-based*													
	Mem0	36.2	37.1	36.2	37.1	–	–	54.5	40.9	38.8	0.376	0.385	0.545	3.23
	A-MEM	43.1	42.2	43.1	42.2	–	–	49.4	58.6	50.1	0.312	0.397	0.561	3.02
	LangMem	39.8	41.3	39.8	41.3	–	–	47.7	47.9	41.1	0.314	0.390	0.564	3.03
	Complete-form Extraction													
	SFT	33.0	32.2	28.3	29.3	3.2	2.1	34.8	32.8	28.3	0.404	0.460	0.277	2.28
	QA-RL-Joint	**65.0**	**64.5**	33.9	34.8	2.7	5.0	45.7	35.8	30.6	0.377	0.366	0.281	2.02
	QA-RL-Swap	35.0	35.9	38.9	38.9	2.7	5.0	48.7	39.3	30.0	0.574	0.483	0.540	2.49
	QA-RL-Frozen	35.5	35.7	38.5	37.3	1.1	4.4	52.5	38.2	35.0	0.585	0.481	0.528	2.72
	QUIET-plain	43.6	36.7	43.5	41.5	25.3	5.8	58.8	44.3	38.8	**0.611**	0.548	**0.549**	**2.81**
	QUIET-diff	48.8	44.7	**47.5**	**44.3**	**32.1**	**15.3**	**60.1**	**49.8**	**42.6**	0.596	**0.553**	0.541	2.73
Qwen3-1.7B	*Retrieval-based*													
	Mem0	36.6	38.5	36.6	38.5	–	–	58.1	52.6	45.2	0.453	0.342	0.547	3.25
	A-MEM	41.3	41.5	41.3	41.5	–	–	61.5	46.1	43.7	0.316	0.422	0.566	3.02
	LangMem	40.3	38.7	40.3	38.7	–	–	49.1	46.3	41.1	0.316	0.414	0.569	2.94
	Complete-form Extraction													
	SFT	32.9	30.0	29.7	28.3	3.3	1.1	12.9	9.0	7.9	0.411	0.464	0.306	2.32
	QA-RL-Joint	**80.4**	**77.7**	39.4	37.3	0.0	1.9	44.1	32.8	32.1	0.375	0.463	0.347	2.29
	QA-RL-Swap	34.5	33.6	39.6	39.0	0.0	1.9	47.0	39.5	31.2	0.557	0.545	0.570	2.70
	QA-RL-Frozen	37.6	33.6	38.0	39.2	2.4	1.6	45.7	36.4	30.0	0.617	0.553	0.591	2.72
	QUIET-plain	43.5	38.2	42.6	39.6	16.7	5.5	55.8	42.9	37.3	**0.619**	0.570	0.574	**2.85**
	QUIET-diff	43.6	39.4	**42.9**	**41.3**	**24.4**	**6.4**	**57.6**	**53.9**	**53.1**	0.595	**0.632**	**0.642**	2.82

4.4 Ablation Study

Answer models. QUIET-diff records the best performance across all answerers on *Fair*PMv2, showing that the advantage of QUIET does not depend on a specific answer model choice (Table 3). Absolute scores vary across answer models, but the ranking QUIET-diff $>$ QUIET-plain $>$ QA-reward RL holds consistently in all conditions. This shows that the preference extraction quality improved through F1 reward RL-training yields consistent benefits across diverse answer models.

Table 3: Ablation study on various answer models on *Fair*PMv2, input context length 32k. Extractor backbone is Qwen3-4B-Instruct. MCQ accuracy (%) is reported.

Method	GPT-5.4-mini	GPT-5.4	Qwen3-14B	Qwen3-32B	Gemini-3.5-flash	Sonnet-4.6
QA-RL-Swap	41.0	42.0	41.9	44.2	43.1	45.4
QA-RL-Frozen	40.3	41.3	41.7	43.3	43.5	43.8
QUIET-plain	47.7	51.1	48.4	46.5	53.5	50.9
QUIET-diff	**49.8**	**53.9**	**49.3**	**50.2**	**57.1**	**54.9**

Reward design. We conduct an ablation study of reward functions based on precision, recall, and many-to-one matching F1. Many-to-one matching allows multiple GT matches for a single extracted bullet. As shown in Figure 4 and Table 4, when recall is used as the reward, the number of extracted bullets expands until the token budget L is exhausted, as the model attempts to maximize recall alone. When precision or many-to-one F1 is used as the reward, reward hacking becomes particularly evident: the model maximizes the reward by concatenating all extracted information into a single bullet, thereby reducing the relevant denominator to one. This violates the intended atomic structure of the preference representation. QUIET uses bipartite F1, achieves the best MCQ accuracy and maintains a number of bullets close to the number of GT preferences.

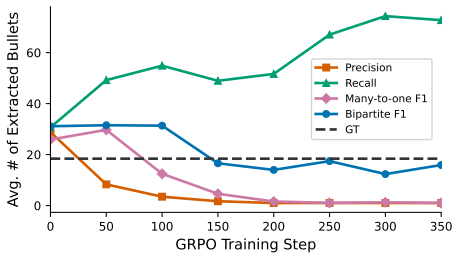


Figure 4: Effect of reward design on the average number of extracted preference bullets on QUIET-diff. Extractor backbone is Qwen3-4B-Instruct.

Table 4: Effect of reward design on *Fair*PMv2 MCQ accuracy, average number of extracted bullets (N), average number of extracted tokens ($|\hat{\mathcal{P}}|$), and semantic F1 score (%) on QUIET-diff, input context length 32k. Extractor backbone model is Qwen3-4B-Instruct.

Reward	MCQ Acc.	N	$\lvert\hat{\mathcal{P}}\rvert$	Semantic F1
Precision	47.7	1.0	763	10.9
Recall	47.2	65.0	1,887	19.9
Many-to-one F1	42.9	1.0	860	19.6
Bipartite F1	**49.8**	17.7	1,621	**44.9**

4.5 Analysis

Effect of dense F1 reward. QUIET computes the F1 reward and updates the model after each individual chunk, providing a dense, per-chunk learning signal. While the QA-reward baseline trains on 18,549 QA instances per epoch, each processing the full conversation, QUIET trains on 5,563 individual chunks. This reduces the per-epoch training context from 747M to 36M tokens, a 21$\times$ reduction, while improving downstream QA performance as shown in Table 5. Dense F1 supervision also outperforms sparse F1 supervision, where the reward and model update occur only at the end of each conversation. It performs better under an equal 36M-token budget and against the sparse baseline trained with 747M tokens. This result suggests that per-chunk updates improve credit assignment by shortening the trajectory between each extraction decision and its learning signal.

Semantic F1 Score Details. According to the extraction statistics in Table 6, extractors trained with QUIET tend to extract a smaller number of bullets, yet still achieve high precision and recall as shown in Table 1. Joint optimization clearly shows its limitation that the QA-reward can be improved through an overfitted answerer, reducing the need for accurate preference extraction. While QA-RL-Frozen substantially improves recall, this gain is largely attributable to extracting an excessive number of bullets, which lowers the precision. QUIET-trained extractors achieve both high precision and recall while maintaining a number of extracted bullets close to the ground truth, as also illustrated in Figure 4, making it the most effective extraction approach.

Table 5: Effectiveness of per-chunk (dense) F1 reward compared to per-conversation (sparse) F1 reward for QUIET-diff on *Fair*PMv2. Extractor backbone is Qwen3-4B-Instruct.

Method	Training context tokens	32k	128k
QA-RL-Joint	747M	37.6	38.3
Sparse F1	747M	48.6	43.6
Sparse F1	36M	46.8	42.6
Dense F1	36M	**49.8**	**44.2**

Table 6: Extraction statistics on the shared context of PMv2 and *Fair*PMv2, 32k input context length. Extractor backbone is Qwen3-4B-Instruct. All values are averaged across personas. N is the average number of extracted bullets, and other metrics are represented in %.

Method	N	Precision	Recall	Semantic F1
QA-RL-Joint	40.2	3.2	7.1	4.4
QA-RL-Frozen	91.1	1.9	10.1	3.2
QUIET-plain	22.4	34.4	**52.8**	40.7
QUIET-diff	17.7	**46.5**	52.5	**47.5**

5 Conclusion

This paper identifies how QA-reward RL can bias preference extractors toward benchmark-specific QA patterns and addresses this issue by introducing a decorrelated benchmark, *Fair*PersonaMem-V2, and a QA-free reward design, QUIET. Experiments show that QUIET consistently improves preference extraction across various benchmarks and extractor model sizes, with gains holding across diverse answer models. However, QUIET relies on an LLM judge for semantic matching, whose errors may introduce noise into the training signal. Moreover, because QUIET focuses solely on extraction, answer-stage functionality such as sensitive-information masking depends on prompt instructions. Future work should develop more reliable semantic rewards and stronger guarantees for answer-stage behavior while preserving the separation between extraction and answering.

AI USE STATEMENT

In this work, we used generative AI tools to construct the *Fair*PersonaMem-V2 benchmark and as components of our experimental pipelines. Specifically, GPT-5.4 was used only for benchmark construction. GPT-5.4-mini was primarily used for semantic F1 reward computation and multiple-choice question answering, while Claude Sonnet 4.6 was used as a plain-text bulletizer and an LLM judge for AlpsBench. Qwen3 and Llama 3.2 models were trained as the preference extractors studied in this work. We also used a range of other generative AI models for ablation studies and robustness comparisons, as detailed in the paper. We did not use generative AI tools to formulate our main claims, and the remaining required disclosure tasks are not applicable to this work. Additionally, we used various generative AI services to assist with software code development, literature search, drafting and refinement of the paper's writing. The authors reviewed all AI-assisted work and take responsibility for the final content of this work, including its text, claims, code, data, results, and other artifacts produced with the aid of generative AI.

ETHICS STATEMENT

*Fair*PersonaMem-V2 and its source benchmark, PersonaMem-V2, consist of synthetic data and do not contain conversations or personal information collected from real individuals. Therefore, the benchmark construction and experiments in this work do not directly raise concerns regarding the privacy of human participants. However, a real-world system based on the proposed preference extraction method may process sensitive information implicitly revealed through user interactions. Extracted preferences could expose private attributes, habits, interests, or other personal information if they were accessed by unauthorized parties, shared beyond their intended purpose, or retained without appropriate safeguards. Deployments of such systems should therefore obtain appropriate user awareness or consent, limit data collection and retention to what is necessary, apply suitable access controls and security protections, and provide users with mechanisms to inspect, correct, or delete extracted preferences. Developers should also evaluate extraction errors and potential biases, as inaccurate or overgeneralized preference profiles could lead to inappropriate personalization. The proposed method is intended to study preference extraction under controlled experimental settings, and its application to real user data requires additional privacy, security, fairness, and legal review appropriate to the deployment context.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the FairPersonaMem-V2 construction procedure, the QUIET training framework, and the experimental setup in Sections 3 and 4. Additional implementation details, ablation studies, training pseudocode, hyperparameters, and the prompts used for preference extraction, MCQ answering, semantic F1 evaluation, and benchmark construction are provided in the appendices. We plan to publicly release the FairPersonaMem-V2 benchmark and the implementation code to facilitate reproduction and further research. The released materials will complement the methodological and experimental details documented in the paper.

REFERENCES

Meta AI. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models, 2024. URL `https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/`.

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.

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

Ge Gao, Alexey Taymanov, Eduardo Salinas, Paul Mineiro, and Dipendra Misra. Aligning llm agents by learning latent preference from user edits. *Advances in Neural Information Processing Systems*, 37:136873–136896, 2024.

Yuren Hao, Shuhaib Mehri, ChengXiang Zhai, and Dilek Hakkani-Tür. User preference modeling for conversational LLM agents: Weak rewards from retrieval-augmented interaction. In *arXiv preprint*, 2026.

Chuanrui Hu, Xingze Gao, Zuyi Zhou, Dannong Xu, Yi Bai, Xintong Li, Hui Zhang, Tong Li, Chong Zhang, Lidong Bing, and Yafeng Deng. EverMemOS: A self-organizing memory operating system for structured long-horizon reasoning. In *Annual Meeting of the Association for Computational Linguistics*, 2026.

Bowen Jiang, Zhuoqun Hao, Young-Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo J. Taylor, and Dan Roth. Know me, respond to me: Benchmarking LLMs for dynamic user profiling and personalized responses at scale. In *Conference on Language Modeling*, 2025a.

Bowen Jiang, Yuan Yuan, Maohao Shen, Zhuoqun Hao, Zhangchen Xu, Zichen Chen, Ziyi Liu, Anvesh Rao Vijjini, Jiashu He, Hanchao Yu, Radha Poovendran, Gregory Wornell, Lyle Ungar, Dan Roth, Sihao Chen, and Camillo Jose Taylor. PersonaMem-v2: Towards personalized intelligence via learning implicit user personas and agentic memory. In *arXiv preprint*, 2025b.

Shuyue Stella Li, Bhargavi Paranjape, Kerem Oktar, Zhongyao Ma, Gelin Zhou, Lin Guan, Na Zhang, Sem Park, Lin Chen, Diyi Yang, Yulia Tsvetkov, and Asli Celikyilmaz. Horizonbench: Long-horizon personalization with evolving preferences. In *COLM 2026 The 2nd Workshop on Lifelong Agents: Learning, Aligning, and Evolving*, 2026. URL https://openreview.net/forum?id=dHJn2aYlLL.

Ziyan Liu, Zhezheng Hao, Yeqiu Chen, Hong Wang, Jingren Hou, Ruiyi Ding, Yongkang Yang, Wence Ji, Wei Xia, and Feng Liu. Meta-cognitive memory policy optimization for long-horizon LLM agents. In *arXiv preprint*, 2026.

Qiyao Ma, Dechen Gao, Rui Cai, Boqi Zhao, Hanchu Zhou, Junshan Zhang, and Zhe Zhao. Personalized rewardbench: Evaluating reward models with human aligned personalization. *Conference on Language Modeling*, 2026a.

Yuchen Ma, Yue Huang, Wenjie Wang, Xiaonan Luo, Xiangliang Zhang, and Stefan Feuerriegel. Synthetic interaction data for scalable personalization in large language models. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 9546–9557. Association for Computing Machinery, 2026b.

Chang Nie, Chaoyou Fu, Yifan Zhang, Haihua Yang, and Caifeng Shan. PersonaVLM: Long-term personalized multimodal LLMs. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

OpenAI. GPT-5.4 [large language model]. https://openai.com/index/introducing-gpt-5-4/, 2026. Large language model, accessed 2026-09-09.

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

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

Alireza Salemi and Hamed Zamani. Lamp-qa: A benchmark for personalized long-form question answering. In *Conference on Empirical Methods in Natural Language Processing*, pp. 1139–1159, 2025.

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. In *arXiv preprint*, 2024.

Yu Wang, Ryuichi Takanobu, Zhiqi Liang, Yuzhen Mao, Yuanzhe Hu, Julian McAuley, and Xiaojian Wu. Mem-α: Learning memory construction via reinforcement learning. In *arXiv preprint*, 2025.

Shujin Wu, May Fung, Cheng Qian, Jeonghwan Kim, Dilek Hakkani-Tur, and Heng Ji. Aligning llms with individual preferences via interaction. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 7648–7662, 2025.

Jianfei Xiao, Xiang Yu, Chengbing Wang, Wuqiang Zheng, Xinyu Lin, Kaining Liu, Hongxun Ding, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Alpsbench: An llm personalization benchmark for real-dialogue memorization and preference alignment. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '26, pp. 3489–3497. Association for Computing Machinery, 2026. ISBN 9798400725999. doi: 10.1145/3805712.3808634. URL https://doi.org/10.1145/3805712.3808634.

Derong Xu, Shuochen Liu, Pengfei Luo, Pengyue Jia, Yingyi Zhang, Yi Wen, Yimin Deng, Wenlin Zhang, Enhong Chen, Xiangyu Zhao, and Tong Xu. Learning how and what to memorize: Cognition-inspired two-stage optimization for evolving memory. In *Annual Meeting of the Association for Computational Linguistics*, 2026a. doi: 10.18653/v1/2026.acl-long.2084.

Jundong Xu, Qingchuan Li, Jiaying Wu, Yihuai Lan, Shuyue Stella Li, Huichi Zhou, Bowen Jiang, Lei Wang, Jun Wang, Anh Tuan Luu, et al. Evoarena: Tracking memory evolution for robust llm agents in dynamic environments. In *arXiv preprint*, 2026b.

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*, 2025. URL https://openreview.net/forum?id=FiM0M8gcct.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z. Pan, Hinrich Schuetze, Volker Tresp, and Yunpu Ma. Memory-R1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. In *Annual Meeting of the Association for Computational Linguistics*, pp. 12805–12825, 2026. doi: 10.18653/v1/2026.acl-long.583. URL https://aclanthology.org/2026.acl-long.583/.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint*, 2025.

Yanyu Yao, Shangze Li, Zhi Zheng, Hui Zheng, Qi Liu, Tong Xu, and Enhong Chen. AtomMem: Building simple and effective memory system for LLM agents via atomic facts. In *arXiv preprint*, 2026.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, and Hao Zhou. Memagent: Reshaping long-context LLM with multi-conv RL-based memory agent. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=k5nIOvYGCL.

Jiaquan Zhang, Chaoning Zhang, Shuxu Chen, Zhenzhen Huang, Pengcheng Zheng, Zhicheng Wang, Ping Guo, Fan Mo, Sung-Ho Bae, Jie Zou, Jiwei Wei, and Yang Yang. Lightweight LLM agent memory with small language models. In *Annual Meeting of the Association for Computational Linguistics*, 2026a.

Qi Zhang, Shen Huang, Chu Liu, Shouqing Yang, Junbo Zhao, Haobo Wang, and Pengjun Xie. DeltaMem: Towards agentic memory management via reinforcement learning. In *arXiv preprint*, 2026b.

Weizhi Zhang, Xinyang Zhang, Chenwei Zhang, Liangwei Yang, Jingbo Shang, Zhepei Wei, Henry Peng Zou, Zijie Huang, Zhengyang Wang, Yifan Gao, Xiaoman Pan, Lian Xiong, Jingguo Liu, Philip S. Yu, and Xian Li. PersonaAgent: Bridging memory and action for personalized LLM agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2026c.

Jihao Zhao, Ding Chen, Zhaoxin Fan, Kerun Xu, Mengting Hu, Bo Tang, Feiyu Xiong, and Zhiyu Li. Inside out: Evolving user-centric core memory trees for long-term personalized dialogue systems. In *Annual Meeting of the Association for Computational Linguistics*, 2026. doi: 10.18653/v1/2026.acl-long.614.

Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. Do LLMs recognize your preferences? evaluating personalized preference following in LLMs. In *International Conference on Learning Representations*, 2025.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Bryan Kian Hsiang Low, and Paul Pu Liang. MEM1: Learning to synergize memory and reasoning for efficient long-horizon agents. In *International Conference on Learning Representations*, 2026.

A Appendix

A.1 Additional Implementation Details

Conversation Processing. All methods process the conversation history incrementally in chunks to avoid exceeding the extraction model's context window. A *chunk* is a contiguous slice of the conversation that fits within a 5,000-token budget (measured by the extraction model's tokenizer). Messages are greedily appended to the current chunk until adding the next message would exceed the budget, at which point the chunk is finalized and a new one begins. If a single message exceeds 5,000 tokens on its own, its content is split into character-estimated sub-messages of approximately 5,000 tokens each, which are then packed normally.

Retrieval-based methods (Mem0, A-Mem, LangMem). Each chunk is submitted to a locally-served extraction model, which processes it differently per framework. **Mem0** distills the chunk into short discrete facts that are upserted into a per-unit vector store. **A-Mem** stores the raw chunk text as a note and runs LLM-based analysis (keywords, context tags, inter-note linking) on it. **LangMem** incrementally extracts and updates structured memory entries, carrying forward the existing memory state across chunks. At answer time, the top-k ($k = 4$) most relevant entries are retrieved by embedding similarity and concatenated as context for the answer model.

Complete-form extraction (SFT, QA-RL, QUIET). For each chunk, the extraction model updates the accumulated memory into a complete updated form. All model checkpoints including QA-reward RL baselines are trained by our implementation. For SFT, QA-reward methods and QUIET-plain use the SFT dataset provided by PMv2, while QUIET-diff uses own dataset tailored to its output format. The QA-RL checkpoints are trained with only the MCQ reward, since the evaluations are all based on MCQ accuracy. The extracted preference memory is bounded by $= 2{,}048$ tokens in all methods.

AlpsBench as a preference extraction evaluation. AlpsBench was originally designed to measure an answer model's personalization ability with the full dialogue history as input. We adapt Task 4 to evaluate the end-to-end performance of a retriever or extractor together with an answer model by replacing the full dialogue history with retrieved or extracted preferences. We evaluate Preference Following (PF), Virtual-Reality Awareness (VRA), Constraint Following (CF), and Emotional Intelligence (EI), and omit Persona Awareness (PA) because it directly provides the target preference and therefore does not test preference construction. Each method first processes the full dialogue history incrementally using the 5,000-token chunks described above. For all methods except SFT and QA-RL-Joint, we fix the answer model to GPT-5.4-mini. For SFT and QA-RL-Joint, we retain the trained model as the answer model because each method is trained to both extract preferences and answer queries. At answer time, complete-form extractors provide the accumulated extracted preferences and the Task 4 query to the assigned answer model. Retrieval-based methods instead provide only the top-4 preferences retrieved for that query and the query itself, not the full dialogue history. This protocol evaluates the combined performance of preference extraction or retrieval and answer generation rather than the answer model's intrinsic personalization ability.

A.2 Benchmark Statistics

Table 7 summarizes the statistics and basic multiple-choice question accuracies of PersonaMem-V2 and *Fair*PersonaMem-V2.

Table 7: Statistics and MCQ accuracy (%) under different preference input conditions for PersonaMem-V2 Jiang et al. (2025b) and *Fair*PersonaMem-V2. Empty accuracy (Empty acc): no preference provided to the answer model. Ground-truth accuracy (GT acc): ground-truth preference set $\mathcal{P}$ provided. The answer model is GPT-5.4-mini.

Benchmark	# personas	# queries	Empty acc	GT acc
PersonaMem-V2	200	5000	33.3	67.5
*Fair*PersonaMem-V2	200	5000	36.4	80.1

A.3 DIFF-FORMAT OPERATIONS AND ENTRIES DETAILS

Edit schema. We represent the extraction state as a dictionary $\hat{\mathcal{P}} : \mathcal{K} \to \mathcal{V}$ from symbolic keys to natural-language preference values. At chunk i, the Diff model outputs a JSON array $\Delta_i = [o_1, \ldots, o_m]$. Each object contains one operation type and a key–value mapping:

$$o_j \in \{\{\texttt{"ADD"}\texttt{:}\ \{\texttt{k:v}\}\},\ \{\texttt{"UPDATE"}\texttt{:}\ \{\texttt{k:v}\}\},\ \{\texttt{"DELETE"}\texttt{:}\ \{\texttt{k:null}\}\}\}. \tag{12}$$

The mapping can contain multiple entries, and the update operator processes them in their JSON order. When the current chunk reveals no preference information, the model outputs the empty array `[]`; this leaves the state unchanged.

Deterministic update semantics. Starting from $\hat{\mathcal{P}}_{i-1}$, U applies the operations in Δ_i sequentially. For every key k and value v, `ADD` inserts $k \mapsto v$ only when $k \notin \hat{\mathcal{P}}$; an `ADD` collision is a no-op. `UPDATE` replaces the value of k only when $k \in \hat{\mathcal{P}}$; an update to a missing key is a no-op. `DELETE` removes k when it exists and otherwise has no effect; its value field is ignored. Consequently, a later operation in the same edit observes the state produced by all preceding operations. Invalid or non-array outputs also leave the state unchanged and receive zero format reward.

Roles of keys and values. Keys provide stable identities for resolving additions, replacements, and deletions across chunks. They do not enter the semantic preference score: after applying an edit, the GRPO reward passes only $\mathrm{values}(\hat{\mathcal{P}}_i)$ to the LLM judge. The judge compares these natural-language value bullets with the gold preferences corresponding to the current chunk, while the number of retained values determines the precision denominator. Thus, key quality affects the transition dynamics, whereas value quality determines semantic coverage and the resulting F1 reward.

A.4 GRPO OPTIMIZATION DETAILS

We optimize f_θ with group relative policy optimization (GRPO) Shao et al. (2024). For each state $s_i = (c_i, \hat{\mathcal{P}}_{i-1})$, the policy samples N_{roll} candidate updates indexed by h. Each candidate produces an updated state $\hat{\mathcal{P}}_{i,h}$ and receives

$$r^{\mathrm{F1}}_{i,h} = \mathrm{SemF1}(\hat{\mathcal{P}}_{i,h}, \mathcal{P}^\star_i). \tag{13}$$

For Diff, we first standardize the F1 and format rewards separately within the candidates sampled for s_i:

$$z^{\mathrm{F1}}_{i,h} = \frac{r^{\mathrm{F1}}_{i,h} - \mu^{\mathrm{F1}}_i}{\sigma^{\mathrm{F1}}_i + \epsilon_1}, \qquad z^{\mathrm{fmt}}_{i,h} = \frac{r^{\mathrm{fmt}}_{i,h} - \mu^{\mathrm{fmt}}_i}{\sigma^{\mathrm{fmt}}_i + \epsilon_1}, \tag{14}$$

where $\mu^{(\cdot)}_i$ and $\sigma^{(\cdot)}_i$ denote the corresponding group mean and standard deviation. The reward supplied to GRPO is then defined as

$$R_{i,h} = \begin{cases} r^{\mathrm{F1}}_{i,h}, & \text{for Plain,} \\ z^{\mathrm{F1}}_{i,h} + \lambda_{\mathrm{fmt}} z^{\mathrm{fmt}}_{i,h}, & \text{for Diff,} \end{cases} \tag{15}$$

where λ_{fmt} controls the contribution of schema validity for Diff. Finally, GRPO standardizes $R_{i,h}$ within the same rollout group to obtain the group-relative advantage:

$$\hat{A}_{i,h} = \frac{R_{i,h} - \mu^R_i}{\sigma^R_i + \epsilon_2}, \tag{16}$$

where μ^R_i and σ^R_i are computed across the candidates sampled for s_i. For Diff, component-wise standardization places the F1 and format rewards on comparable scales before aggregation, while the final GRPO standardization accounts for the scale and variance of the aggregated reward. For Plain, the same final standardization is applied directly to the F1 reward. The standard GRPO objective then increases the likelihood of updates with positive group-relative advantage. Because this optimization is performed at each sequential state, credit is assigned directly to the preference update that produced it rather than through a later answer trajectory. No answer tokens are generated during QUIET training; after extraction, any answer model can consume the final state $\hat{\mathcal{P}}_K$ as defined in Section 3.1.

A.5 Formal definitions of progressive comparison

Here we provide formal definitions of four progressive comparison configurations at table 1, QA-RL-Joint, QA-RL-Swap, QA-RL-Frozen, and QUIET. $\mathcal{H}$, q, and $y^\star$ denote conversation, query, and correct answer of the givne QA set, respectively. $\mathcal{H}_i$ denotes the conversation up to the i-th chunk.

Table 8: Training and inference configurations. θ denotes the trainable parameters, and ϕ denotes frozen parameters from a weight distinguished from θ. $\mathbb{I}[\hat{y} = y^\star]$ is the QA correctness reward, and $\mathrm{SemF1}(\hat{\mathcal{P}}, \mathcal{P}^\star)$ is the semantic F1 reward.

QA-RL-Joint	**QA-RL-Swap**
Train. $\hat{y} = g_\theta(f_\theta(\mathcal{H}), q)$ and $r^{QA} = \mathbb{I}[\hat{y} = y^\star]$. GRPO updates the shared parameters θ. *Inference.* $\hat{y} = g_\theta(f_\theta(\mathcal{H}), q)$.	*Train.* $\hat{y} = g_\theta(f_\theta(\mathcal{H}), q)$ and $r^{QA} = \mathbb{I}[\hat{y} = y^\star]$. GRPO updates the shared parameters θ. *Inference.* $\hat{y} = g_\phi(f_\theta(\mathcal{H}), q)$, where $\phi \neq \theta$.
QA-RL-Frozen	**QUIET (Ours)**
Train. $\hat{y} = g_\phi(f_\theta(\mathcal{H}), q)$ and $r^{QA} = \mathbb{I}[\hat{y} = y^\star]$. GRPO updates only θ; ϕ is frozen. *Inference.* $\hat{y} = g_\phi(f_\theta(\mathcal{H}), q)$.	*Train.* $\hat{\mathcal{P}}_i = f_\theta(\mathcal{H}_i)$ and $r_i^{F1} = \mathrm{SemF1}(\hat{\mathcal{P}}_i, \mathcal{P}_i^\star)$. GRPO updates only θ; no answerer is invoked. *Inference.* $\hat{y} = g_\phi(f_\theta(\mathcal{H}), q)$.

A.6 Additional Ablation Studies

Format-reward weight. Even without a format reward ($\lambda_{\mathrm{fmt}} = 0.0$), the format-invalid rate remains low at 2.14%, indicating that the F1 reward already provides guidance toward format compliance. Nevertheless, MCQ accuracy is limited to 45.4%, suggesting that the model struggles to extract structured semantic information without the format reward. Adding a format reward with small weight ($\lambda_{\mathrm{fmt}} = 0.1$) eliminates format-invalid outputs and improves MCQ accuracy to 46.8%. At $\lambda_{\mathrm{fmt}} = 0.2$, the model achieves the highest semantic F1 score (45.9%) and MCQ accuracy (49.8%), so we use this value as the default for QUIET. Larger format-reward weights reduce both semantic F1 and MCQ accuracy, suggesting that excessive format reward overemphasizes format compliance relative to semantic signals.

Table 9: Effect of the format-reward weight λ_{fmt} for QUIET-diff with a Qwen3-4B extractor. We evaluate with a 32k input context length on *Fair*PMv2 and report average number of extracted preference bullets (N), along with format invalid rate, semantic F1 and MCQ accuracy in percent.

λ_{fmt}	N	Format Invalid	F1	MCQ
0.0	14.1	2.14	42.8	45.4
0.1	15.0	0.00	41.0	46.8
0.2	18.3	0.00	**45.9**	**49.8**
0.5	16.2	0.00	43.5	48.1
1.0	14.4	0.00	42.9	46.3

Semantic F1 Judge Robustness. Semantic F1 score exhibits broadly consistent trends across judge models. Both QUIET variants score substantially above the QA-reward baselines under every judge in Table 10. Absolute F1 values vary because judge models differ in how strictly they determine semantic matches under the same prompt. Semantic F1 score is also efficient because one judge call evaluates all extracted preferences for a persona, covering the preferences required for 25 MCQ items on average in the case of PMv2.

A.7 Pseudo Code for model training

Algorithm 1 summarizes the recurrent GRPO procedure used to train QA-RL-Joint. The same policy acts as both the preference extractor and the answerer. Each sampled trajectory first compresses the conversation chunk by chunk into a recurrent preference state and then answers the MCQ using only the final extracted preferences. The terminal QA reward is normalized across trajectories sampled for the same example and broadcast to every generated token in the corresponding trajectory, thereby assigning credit to both the intermediate preference-state updates and the final answer.

Table 10: Semantic preference F1 (%) under different judge models. All methods use a Qwen3-4B-Instruct extractor backbone with a 32k-token input context. The extracted preferences from QA-RL-Joint, QA-RL-Frozen, QUIET-plain, and QUIET-diff are fixed; only the judge model used for bipartite semantic matching changes.

Judge model	QA-RL-Joint	QA-RL-Frozen	QUIET-plain	QUIET-diff
GPT-5.4-mini	4.35	3.16	40.71	**45.89**
GPT-5.4	8.41	13.25	44.74	**50.91**
Qwen3-14B	5.67	2.36	28.12	**48.46**
Qwen3-32B	7.71	2.53	36.86	**48.48**
Gemini-3.5-Flash	1.95	3.53	25.08	**42.01**
Sonnet-4.6	8.60	10.97	47.50	**57.56**

Algorithm 1 Recurrent QA-reward GRPO training for QA-RL-Joint

Require: Training set $\mathcal{D} = \{(\mathcal{H}, q, y^\star)\}$, policy π_θ, reference policy π_{ref}, rollout policy $\pi_{\theta_{\text{old}}}$, group size G, chunk budget K, clipping threshold ϵ, numerical constant ϵ_{adv}, KL coefficient β

for each minibatch $\mathcal{B} \subset \mathcal{D}$ **do**
  for all $(\mathcal{H}, q, y^\star) \in \mathcal{B}$ **do**
    Split $\mathcal{H}$ into ordered chunks $(c_1, \ldots, c_T)$, where $T \le K$
    for $h = 1, \ldots, G$ **do**
      Initialize the recurrent preference state $\hat{\mathcal{P}}_{0,h} \leftarrow \varnothing$
      for $i = 1, \ldots, T$ **do**
        Construct $s_{i,h} \leftarrow (q, \hat{\mathcal{P}}_{i-1,h}, c_i)$
        Sample a complete updated preference state $\tilde{\mathcal{P}}_{i,h} \sim \pi_{\theta_{\text{old}}}(\cdot \mid s_{i,h})$
        $\hat{\mathcal{P}}_{i,h} \leftarrow \textsc{StripThinking}(\tilde{\mathcal{P}}_{i,h})$
      end for
      Construct the answer state $s_h^{\text{ans}} \leftarrow (q, \hat{\mathcal{P}}_{T,h})$
      Sample $\hat{y}_h \sim \pi_{\theta_{\text{old}}}(\cdot \mid s_h^{\text{ans}})$
      Assign terminal reward $R_h \leftarrow \mathbb{I}[\textsc{BoxedAnswer}(\hat{y}_h) = y^\star]$
    end for
    $\mu_R \leftarrow G^{-1} \sum_{h=1}^{G} R_h, \quad \sigma_R \leftarrow \sqrt{G^{-1} \sum_{h=1}^{G} (R_h - \mu_R)^2}$
    for $h = 1, \ldots, G$ **do**
      $\hat{A}_h \leftarrow (R_h - \mu_R)/(\sigma_R + \epsilon_{\text{adv}})$
      Assign $\hat{A}_h$ to every generated extraction and answer token in trajectory h
    end for
  end for
  $\rho_t(\theta) \leftarrow \exp(\log \pi_\theta(a_t \mid s_t) - \log \pi_{\theta_{\text{old}}}(a_t \mid s_t))$
  $\mathcal{J}(\theta) \leftarrow \mathbb{E}_t\Big[\min\Big(\rho_t(\theta)\hat{A}_t, \text{clip}(\rho_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\Big)\Big] - \beta\, \mathbb{E}_t[D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{ref}})]$
  Update θ by maximizing $\mathcal{J}(\theta)$
end for

In our implementation, $G = 8$, each conversation chunk contains at most $5{,}000$ tokens, $K = 8$, $\epsilon = 0.2$, and $\beta = 0.001$. Both preference updates and the final answer are capped at $2{,}048$ generated tokens. We use temperature 1.0 and top-$p = 1.0$ for training rollouts, and optimize the model with a learning rate of 10^{-6} without an entropy bonus.

Algorithm 2 gives the corresponding training procedure for QUIET-plain and QUIET-diff. Both variants optimize the extractor directly at every chunk using the cumulative preference target. They differ only in how an output represents and updates the running preference state and in whether schema validity contributes to the reward.

For both formats, the bipartite semantic score assigns each predicted preference entry to at most one gold preference and counts each gold preference at most once. If $t_{i,h}$ distinct gold preferences are matched, the reward is $r_{i,h}^{\text{F1}} = 2t_{i,h}/(|\hat{\mathcal{P}}_{i,h}| + |\mathcal{P}_i^\star|)$. Plain directly supplies this score to GRPO. Diff first standardizes the semantic F1 and binary format scores separately within the G candidates and

Algorithm 2 Unified recurrent GRPO training for QUIET-plain and QUIET-diff

Require: Persona trajectories $\mathcal{D} = \{(c_{1:K}, \mathcal{P}^{\star}_{1:K})\}$, format $m \in \{\textsc{Plain}, \textsc{Diff}\}$, policy π_θ, reference policy π_{ref}, rollout policy $\pi_{\theta_{\mathrm{old}}}$, group size G, format weight λ_{fmt}, state budget L, clipping threshold ϵ, numerical constants ϵ_1, ϵ_2, KL coefficient β

for each minibatch of persona trajectories **do**
  Initialize one running preference state $\hat{\mathcal{P}}_0 \leftarrow \varnothing$ for each persona
  for $i = 1, \ldots, K$ **do**
    Construct state $s_i \leftarrow (c_i, \hat{\mathcal{P}}_{i-1})$
    for $h = 1, \ldots, G$ **do**
      Sample output $o_{i,h} \sim \pi_{\theta_{\mathrm{old}}}(\cdot \mid s_i)$
      if $m = \textsc{Plain}$ **then**
        $\hat{\mathcal{P}}_{i,h} \leftarrow \textsc{ParsePlainState}(o_{i,h})$ ▷ rewrite the complete state
      else
        $\Delta_{i,h} \leftarrow \textsc{ParseJSONOperations}(o_{i,h})$
        $\hat{\mathcal{P}}_{i,h} \leftarrow \textsc{Trim}(U(\hat{\mathcal{P}}_{i-1}, \Delta_{i,h}), L)$ ▷ apply only the predicted changes
      end if
      $r^{\mathrm{F1}}_{i,h} \leftarrow \mathrm{SemF1}_{\mathrm{bip}}(\hat{\mathcal{P}}_{i,h}, \mathcal{P}^{\star}_i)$
      if $m = \textsc{Diff}$ **then**
        $r^{\mathrm{fmt}}_{i,h} \leftarrow \mathbb{I}[o_{i,h} \text{ is a valid JSON operation array}]$
      end if
    end for
    if $m = \textsc{Plain}$ **then**
      $R_{i,h} \leftarrow r^{\mathrm{F1}}_{i,h}$ for all h
    else
      Compute group statistics $(\mu^{\mathrm{F1}}_i, \sigma^{\mathrm{F1}}_i)$ and $(\mu^{\mathrm{fmt}}_i, \sigma^{\mathrm{fmt}}_i)$ over $h = 1, \ldots, G$
      $z^{\mathrm{F1}}_{i,h} \leftarrow (r^{\mathrm{F1}}_{i,h} - \mu^{\mathrm{F1}}_i)/(\sigma^{\mathrm{F1}}_i + \epsilon_1)$ for all h
      $z^{\mathrm{fmt}}_{i,h} \leftarrow (r^{\mathrm{fmt}}_{i,h} - \mu^{\mathrm{fmt}}_i)/(\sigma^{\mathrm{fmt}}_i + \epsilon_1)$ for all h
      $R_{i,h} \leftarrow z^{\mathrm{F1}}_{i,h} + \lambda_{\mathrm{fmt}} z^{\mathrm{fmt}}_{i,h}$ for all h
    end if
    Compute (μ^R_i, σ^R_i) over $\{R_{i,h}\}_{h=1}^G$
    $\hat{A}_{i,h} \leftarrow (R_{i,h} - \mu^R_i)/(\sigma^R_i + \epsilon_2)$ for all h
    Assign $\hat{A}_{i,h}$ to every generated token of $o_{i,h}$
    $\rho_{i,h,t}(\theta) \leftarrow \exp(\log \pi_\theta(o_{i,h,t} \mid s_{i,h,t}) - \log \pi_{\theta_{\mathrm{old}}}(o_{i,h,t} \mid s_{i,h,t}))$
    $\mathcal{J}_i(\theta) \leftarrow \mathbb{E}_{h,t}\Big[\min\Big(\rho_{i,h,t}(\theta)\hat{A}_{i,h}, \mathrm{clip}(\rho_{i,h,t}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{i,h}\Big)\Big] - \beta\, \mathbb{E}_{h,t}[D_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\mathrm{ref}})]$
    Update θ by maximizing $\mathcal{J}_i(\theta)$
    $\hat{\mathcal{P}}_i \leftarrow \hat{\mathcal{P}}_{i,1}$ ▷ propagate the first rollout to the next chunk
  end for
end for

combines them using λ_{fmt}; the standard GRPO estimator subsequently standardizes the combined reward again. In both variants, the first sampled candidate determines the running preference state supplied to the next chunk, while all G candidates contribute to the current chunk's policy update. The subsequent clipped GRPO and KL-regularized policy update is identical to that of QA-RL-Joint; only the reward definition and the granularity of credit assignment differ.

A.8 Extraction and Evaluation Prompts

Table 11: Extraction, MCQ answer, and semantic F1 judge prompt templates.

Prompt	Template
Extraction (baseline)	You are presented with a conversation history between the current user and the chatbot. Your task is to carefully analyze this conversation history and extract information about the user's persona and preferences that are revealed or indicated explicitly or implicitly through their interactions, responses, and dialogue patterns. Focus on identifying personas and preferences of the current user, focusing on those implicitly indicated in the user-chatbot conversation histories. Update the memory by retaining all relevant, still up-to-date details from the previous memory while adding any new, useful persona and preference information discovered in the current conversation section. The memory should be clean, standalone, and in English. Do NOT include any other texts in your response. <previous_memory> {*memory*} </previous_memory> <conversation_section> {*chunk*} </conversation_section> Updated memory:
Extraction (plain)	[Same task description as baseline, with the following modification:] Produce the FULL UPDATED MEMORY: keep every still-valid preference from the current memory, add any new preference, revise a line when the section refines or contradicts it, and drop a line when the user explicitly asks to forget it. Output rules: one preference per line starting with "–"; each line is a single, self-contained, deduplicated preference in English; output ONLY the memory lines. <memory> {*memory*} </memory> <section> {*chunk*} </section> Updated memory:
Extraction (diff)	[Same task description as baseline, with the following modification:] For each extracted preference: ADD if not already covered, UPDATE if it contradicts or refines an existing entry, DELETE if the user asks to forget it. Do NOT invent a new key for a concept already tracked under a different name. Output Format MUST be a valid JSON array: `[{"ADD": {"key": "val"}}, {"UPDATE": {"key": "val"}}, {"DELETE": {"key": null}}]` Output an empty array `[]` if there are no preferences in the chunk. <memory> {*memory*} </memory> <section> {*chunk*} </section> Output your JSON array below:
Answer prompt	You are presented with a user query and previous memory about the user's persona and preferences. Please answer the user query based on the previous memory and provide a personalized response. Put your final answer in \boxed{}. The memory and user query below may contain sensitive personal information. You MUST mask out or avoid direct exposure of these in the response. Only choose a specific answer if memory contains a preference that *directly and explicitly* matches the topic of the query. If no direct match exists, choose the generic answer. <user_query> {*prompt*} </user_query> <user_memory> {*memory*} </user_memory> Your answer:
Semantic F1 judge	Perform semantic matching of EXTRACTED bullets to GROUND TRUTH (GT) preferences. A match is valid only when the extracted bullet preserves the essential meaning and distinguishing attributes of the GT item. If there is clear match, assign only one best matching GT index (1-based) to the extracted bullet. Do not match based on topical similarity, partial overlap, or generic statements. If a bullet does not clearly correspond to a GT item, output 0. Ground truth items: {*items*} Extracted bullets: {*bullets*} Respond with ONLY an array of length {*n_bullets*}. No explanation.

A.9 *Fair*PERSONAMEM-V2 CONSTRUCTION PROMPTS

Table 12: Prompt templates used in *Fair*PersonaMem-V2 benchmark construction.

Prompt	Template
Choice generation	Given this user persona: {*persona*} and this specific user preference: {*preference*} You are creating a multiple-choice benchmark. There is no question yet — you will design the four answer options first. First, break down the user preference into category (medical history, music interest, hobby, etc), attribute, and relation (like, dislike, experienced, etc). Then make up two decoy preferences by altering the attribute. The decoys must sound realistic, be highly related to the original, and fall into the same category naturally. Generate four different, one-to-three sentence answers: (1) `correct`: personalized to the user's true preference. (2) `decoy1`: personalized with the first decoy. (3) `decoy2`: personalized with the second decoy. (4) `generic`: identical in structure with `correct` but generic, suitable for anyone. All four answers MUST be about the same situation/scenario so that a single natural question could plausibly elicit any one of them. Each answer must have the same tone and length.
Query generation	You are given three candidate user preferences that all belong to the same category: (1) {*true_pref*}, (2) {*decoy1_pref*}, (3) {*decoy2_pref*}. Write a first-person, natural question that could plausibly be asked by this user to the chatbot. The question should invite recommendations, ideas, advice, or explanations. The question must be NEUTRAL among ALL THREE preferences: phrased so that a personalized answer built on ANY ONE of the three would be an equally plausible response. Frame the question at the shared, higher-level topic. Do NOT mention or allude to any of the three preferences, and do NOT refer to "my preferences" or "what I like."