

Leaving the Past Behind: Breaking Personalization Inertia in Long-Term LLM Agents

Wenhao Yuan¹, Chenchen Lin¹, Jian Chen¹, Weijie Liu¹, Jinfeng Xu¹,
Shuo Yang¹, Edith Cheuk Han Ngai^{1,*}

¹The University of Hong Kong,
wenhao.yuan@connect.hku.hk, chngai@eee.hku.hk

Abstract

LLM agents rely on long-term interaction histories to tailor their responses to individual users. However, historical personalization signals may become stale or misaligned with expectations implied by the current context, leading agents to follow outdated preferences during reasoning. We refer to this failure mode as *personalization inertia*, which shifts the agent’s behavior away from the user’s current intent. Existing methods mainly optimize final responses or memory retrieval, leaving such failures underexplored. In this paper, we propose **Personalization Inertia Verification and Regulation (PIVOT)**, a framework for mitigating personalization inertia in long-term interactions with agents. Concretely, we construct a trajectory-independent evidence condition as a diagnostic reference, model prefix-evidence consistency to assess compatibility with the current personalization evidence, and learn a rollback policy to determine whether and where to intervene. The selected intervention triggers local repair, preserving the reliable prefix while regenerating only the potentially inertia-affected suffix under evidence-guided constraints. Experiments on benchmark datasets demonstrate consistent gains in personalization accuracy and trajectory faithfulness with targeted intervention overhead.

1 Introduction

Large Language Models (LLMs) are increasingly deployed as personalized agents that engage with users over extended interactions, maintain user-specific memories, and adapt responses to evolving preferences and feedback (Packer et al., 2023; Zhong et al., 2024). Long-term personalization typically relies on historical interaction records, including past requests, retrieved memories, and feedback, to infer user intent and generate context-aware responses (Salemi et al., 2024; Li et al.,

*Corresponding author.

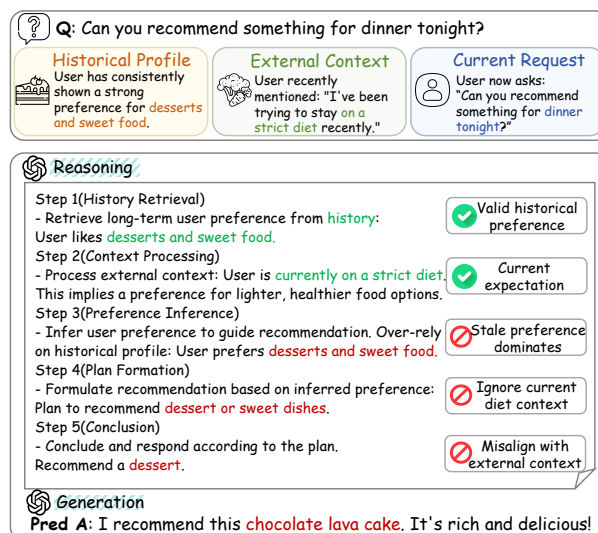


Figure 1: Illustration of personalization inertia in long-term LLM agents. Although external contexts indicate that the user is trying to follow a strict diet, the agent retrieves and relies on an outdated preference for desserts and sweet food. This stale preference propagates via intermediate reasoning steps and ultimately produces a response that conflicts with the current expectation.

2025). However, such historical signals may become outdated or misaligned with current expectations, leading agents to rely on stale memories or obsolete preferences even when the latest evidence suggests otherwise (Wu et al., 2025a). We refer to such scenarios as *personalization inertia*, as illustrated in Figure 1. A historical preference conflict with the current expectation may still be retrieved, used to infer the preference state, and propagated into subsequent reasoning and acting, ultimately leading to a response that violates the latest intent (Yao et al., 2023b; Shinn et al., 2023; Lyu et al., 2023). Further, the final response does not reveal where personalization inertia enters. This lack of localization is particularly harmful in long-term interactions, where inertia-driven states may be written back into memory and reused (Xu et al., 2026), thereby reinforcing outdated user models

and causing persistent behavioral drift.

In personalized LLM agents, existing methods primarily improve personalization through memory retrieval (Pan et al., 2025; Yuan et al., 2025), profile construction (Zhang et al., 2025a), preference modeling (Zhang et al., 2025b), or response-level alignment (Wu et al., 2025b; Zhao et al., 2025a; Wang et al., 2024). While these methods enable agents to exploit history, they generally assume retrieved or stored personalization signals remain valid once incorporated into the decision process. Recent work has studied preference drift (Zhao et al., 2025b; Tan et al., 2025), memory updating (Yan et al., 2025; Chhikara et al., 2025), and forgetting-aware personalization, but their impacts on intermediate decision processes in agents remain underexplored. Consequently, stale memories or outdated patterns may still propagate before the agent commits to a final response or updates memory (Xu et al., 2026). Prevailing correction strategies, such as response rewriting or memory editing (Shinn et al., 2023; Madaan et al., 2023), provide limited support for repairing personalization inertia at the granularity of trajectory segments. This motivates inertia-aware mechanisms that verify intermediate decisions against current contexts while preserving useful historical evidence, raising a key question: *How can personalized LLM agents detect, localize, and regulate personalization inertia during intermediate trajectory generation under evolving user states?*

To overcome these limitations, we propose **Personalization Inertia Verification and Regulation (PIVOT)**, a novel framework for consistency verification and rollback repair before response commitment. PIVOT adopts a trajectory-independent evidence condition from current contexts, user requests, and historical memories, providing a stable diagnostic reference before trajectory inspection. It then introduces prefix consistency modeling to estimate whether each partial trajectory remains compatible with the current evidence condition and identify potential inertia emergence points through local consistency drops, based on which PIVOT constructs rollback-priority candidates and learns a lightweight rollback policy that determines whether and where intervention should occur. Finally, PIVOT preserves reliable trajectory prefixes and regenerates only the potentially inertia-affected suffix under evidence-guided constraints, enabling minimal and targeted correction. Our key contributions are summarized as follows:

- We identify the overlooked issue of personalization inertia in long-term LLM agents, where stale personalization signals propagate via trajectories and contaminate memory.
- We introduce PIVOT, a novel framework that verifies prefix-evidence consistency and enables policy-guided rollback repair before response commitment.
- We conduct extensive experiments on multiple datasets to validate the effectiveness of PIVOT in mitigating personalization inertia.

2 Related Work

2.1 Long-Term Personalization in Agents

Personalization has become an important perspective for adapting LLM agents to individual users. Existing work studies personalized generation from histories or profiles, showing that historical signals can improve response relevance, style consistency, and preference alignment (Salemi et al., 2024; Kumar et al., 2024; Kirk et al., 2024). Subsequent methods improve user modeling through profile-based prompting (Wu et al., 2024), few-shot personalization (Kim and Yang, 2025), or preference modeling (Poddar et al., 2024; Gao et al., 2024), and difference-aware user representations (Qiu et al., 2025). Beyond static profiles, recent personalized agents incorporate long-term interaction histories and memory mechanisms to support continuity across dialogue, recommendation, and tool-use settings (Jiang et al., 2025; Shang et al., 2025; Hao et al., 2025). These works demonstrate the importance of exploiting historical information, especially when preferences are implicit or distributed over multiple interactions.

Despite these advances, most long-term personalization methods focus on constructing or exploiting user-specific signals. In agentic settings, when user preferences evolve, outdated memories may affect the reasoning before the final response. Existing personalization methods provide limited mechanisms for checking the compatibility of historical signals and the current context.

2.2 Trajectory Verification and Repair

Recent work has studied to verify and improve multi-step reasoning trajectories rather than outputs. For LLM reasoning, verifier-based methods train or prompt models to judge intermediate steps or final solutions (Lightman et al., 2024). Other

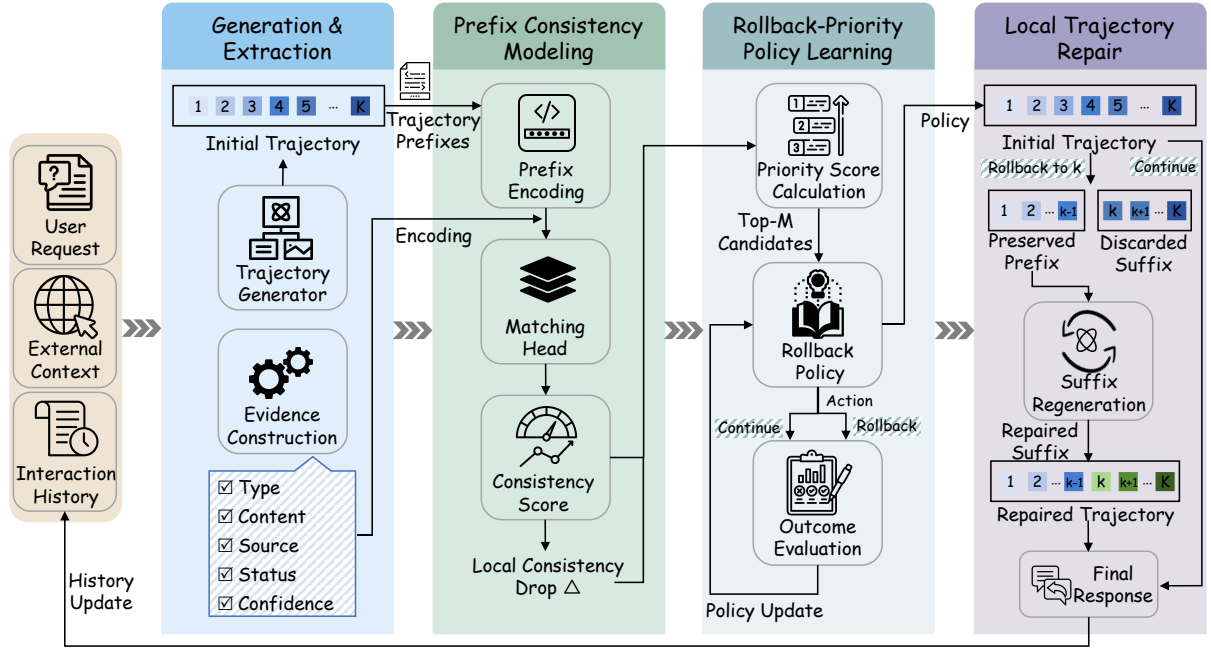


Figure 2: An overview of our PIVOT framework. This diagram illustrates the overall procedure from evidence construction, prefix-evidence consistency verification, policy-guided rollback strategies, and trajectory repair.

approaches improve reliability through search or aggregation over multiple trajectories, such as self-consistency (Wang et al., 2023), tree-structured deliberation (Yao et al., 2023a), and verifier-guided reasoning (Li et al., 2023). In agentic settings, trajectory reliability is often addressed through external feedback or critic-based revision, where agents use environment signals or self-evaluation to revise plans and actions (Gou et al., 2024; Huang et al., 2024; Chen et al., 2024), showing that intermediate reasoning and action traces provide useful signals for diagnosing and improving behavior.

Despite the progress, existing verification and repair methods are task-centric or outcome-driven, checking whether a reasoning step is executable or logically valid (Yuan et al., 2026), but fail to validate whether the step remains compatible with a user’s current preference state under long-term personalization. Moreover, many repair mechanisms regenerate whole responses or rerank sampled trajectories, rather than localizing where an outdated user signal first affects the trajectory.

3 Methodology

In this section, we introduce PIVOT with the framework shown in Figure 2. We formulate personalization inertia and construct an evidence condition in § 3.1, introduce prefix consistency modeling in § 3.2, present a rollback-priority policy in § 3.3,

and describe the repair mechanism in § 3.4.

3.1 Inertia-Aware Personalized Formulation

We consider personalized LLM agents in long-term interactions, where the user’s request is addressed through a sequence of intermediate trajectory steps rather than a single independent generation (Li et al., 2025; Yuan et al., 2026). Although historical interactions improve personalization, they may induce *personalization inertia*: a trajectory’s tendency to follow stale historical preferences despite conflicting current expectations, causing failures before the final response.

At interaction round t , the agent receives external context x_t , user request u_t , and the interaction history H_t . Different from historical preferences already summarized in H_t , the user’s current expectation may be implicitly expressed in the external context x_t and has not yet been incorporated into H_t . Therefore, a mismatch can arise when the preference inferred from H_t conflicts with the expectation indicated by x_t . In such cases, the agent may still follow the historical preference during trajectory generation, causing its response to deviate from the user’s current expectation. Given (x_t, u_t, H_t) , the personalized agent generates an initial structured trajectory $\tau_t \sim \pi_{\text{gen}}(\cdot \mid x_t, u_t, H_t)$, represented as

$$\tau_t = \{y_t^{(1)}, y_t^{(2)}, \dots, y_t^{(K_t)}\}. \quad (1)$$

where K_t is the number of intermediate steps, and each step $y_t^{(k)}$ denotes a semantic operation such as intent understanding, memory retrieval, preference inference, or planning. To provide an operational anchor for inertia-aware inspection, we construct a trajectory-independent evidence condition before evaluating the generated trajectory:

$$\delta_t = \text{Evi}(x_t, u_t, H_t), \quad (2)$$

which captures the current expectation implied by x_t and contrasts it with preferences summarized in H_t , assigning provisional statuses according to source priority, recency, explicit corrections, and conflicts with the current expectation¹. We represent δ_t as structured records $\delta_t = \{d_{t,j}\}_{j=1}^{N_t}$, $d_{t,j} = (\text{type}_j, \text{content}_j, \text{source}_j, \text{status}_j, \text{conf}_j)$, where status_j and conf_j denote the provisional status and confidence. More details are provided in Appendix A. For instance, current constraints are treated as higher-priority evidence over historical preferences, and recent feedback marks earlier memories as potentially contradicted.

3.2 Prefix Consistency Modeling

Given the trajectory τ_t and evidence condition δ_t , we inspect whether each partial trajectory remains compatible with the current personalization evidence. Since personalization failures may originate before the final response, we evaluate prefixes rather than only the output. For a trajectory τ_t , the prefix up to step k is defined as $P_t^{(k)} = \{y_t^{(1)}, y_t^{(2)}, \dots, y_t^{(k)}\}$. Such inspections provide diagnostic signals for identifying whether the partial decision process remains consistent with current evidence.

To obtain an operational representation, we serialize each trajectory step as $\text{Ser}(y_t^{(k)})$, which records the step type, evidence, output, and dependencies on memories or preferences. We encode the representation of each step- k as $e_t^{(k)} = \text{StepEnc}(\text{Ser}(y_t^{(k)}), x_t, u_t, H_t)$. The prefix representation is then obtained by $h_t^{(k)} = \text{Pool}(e_t^{(1)}, e_t^{(2)}, \dots, e_t^{(k)})$, which summarizes only the information available up to step k without considering future steps. We also encode the inertia-aware evidence condition into $h_\delta^t = \text{RefEnc}(\text{Ser}(\delta_t), u_t)$. Given $h_t^{(k)}$ and h_δ^t , we compute prefix-evidence consistency through a lightweight matching head:

$$c_t^{(k)} = \text{Sigmoid}(\text{MLP}_c([h_t^{(k)}; h_\delta^t; |h_t^{(k)} - h_\delta^t|])), \quad (3)$$

¹ δ_t is constructed before trajectory inspection and does not access τ_t or final responses, avoiding diagnostic leakage.

where MLP_c outputs a scalar logit, $|\cdot|$ denotes element-wise absolute difference, and $\text{Sigmoid}(\cdot)$ maps the logit into $[0, 1]$. The consistency head is trained with weak prefix-level labels derived from the status-annotated evidence records in δ_t , where prefixes relying on stale, overridden, or contradicted evidence are labeled as inconsistent and labels are confidence-weighted to mitigate noise; details are provided in Appendix B. A larger $c_t^{(k)}$ indicates stronger compatibility between the prefix and the evidence condition. We further compute the local consistency drop by $\Delta_t^{(k)} = c_t^{(k-1)} - c_t^{(k)}$, $k \geq 2$. A positive $\Delta_t^{(k)}$ indicates that adding step k decreases compatibility with the current evidence, serving as a diagnostic signal rather than causal proof, as consistency scores may vary with step characteristics and representation uncertainty.

3.3 Rollback-Priority Policy Learning

The prefix consistency model provides compatibility scores $\{c_t^{(k)}\}_{k=1}^{K_t}$ and local consistency drops $\{\Delta_t^{(k)}\}_{k=2}^{K_t}$. These signals indicate where the trajectory becomes less aligned with the evidence condition δ_t , but do not directly determine whether or where to intervene. A consistency drop may reveal that an inconsistency has become visible, while the useful intervention point may lie earlier in the trajectory. We therefore construct rollback-priority scores to propose candidate repair points, and then learn a lightweight policy to select the final intervention action. For each k , we combine the prefix-evidence compatibility and the consistency decrease via a hazard-inspired priority score:

$$r_t^{(k)} = 1 - c_t^{(k)} \exp(-\eta[\Delta_t^{(k)}]_+), \quad (4)$$

where $[\Delta_t^{(k)}]_+ = \max(0, \Delta_t^{(k)})$, $\eta \geq 0$ controls the influence of the local consistency decrease, and $\Delta_t^{(1)} = 0$. The score increases when the prefix compatibility is low or when adding step k causes a larger positive consistency drop. Thus, $r_t^{(k)}$ provides a bounded and monotonic candidate prior without introducing an additional trainable rollback scorer. We then construct a compact candidate set $\mathcal{I}_t = \text{TopM}(\{r_t^{(k)}\}_{k=1}^{K_t})$, where $\text{TopM}(\cdot)$ selects the M steps with the highest priority scores, restricting intervention to a small number of semantic trajectory steps, rather than arbitrary token positions or all intermediate operations.

The priority score proposes where repair may be useful, but it does not decide whether repair should

be performed. The final intervention decision is made by a rollback policy. The policy observes $o_t = (x_t, u_t, H_t, \tau_t, \delta_t, \{c_t^{(k)}, \Delta_t^{(k)}, r_t^{(k)}\}_{k=1}^{K_t}, \mathcal{I}_t)$, and samples an action $a_t \sim \pi_\theta(\cdot | o_t)$ from the action space $\mathcal{A}_t = \{\text{continue}\} \cup \{\text{rollback_to } k \mid k \in \mathcal{I}_t\}$. The action `continue` returns the original trajectory, while `rollback\_to` k triggers local repair from semantic step k , as described in § 3.4. This design separates candidate proposal from intervention control: $r_t^{(k)}$ proposes plausible repair points, while π_θ learns whether rollback is worthwhile under the current interaction context.

We optimize the rollback policy by comparing intervention actions under the same observation o_t . For each $a \in \mathcal{A}_t$, let $\hat{\tau}_t(a)$ denote the trajectory obtained by applying a . If $a = \text{continue}$, then $\hat{\tau}_t(a) = \tau_t$; if $a = \text{rollback_to } k$, then $\hat{\tau}_t(a)$ is the trajectory repaired from step k by the repair module. Each action is assigned an outcome score:

$$Q_t(a) = G_t(\hat{\tau}_t(a), x_t, u_t, \delta_t, H_t) - \lambda_c C_t(a), \quad (5)$$

where $C_t(\cdot)$ denotes the intervention cost, and $G_t(\cdot)$ is an evidence-grounded outcome evaluator that compares candidate trajectories under the same observation o_t , which can be instantiated via rule-based checks or an LLM-as-judge (details in Appendix C). Since all candidate actions are evaluated under the same observation, we normalize outcomes within the action group as follows:

$$A_t(a) = \frac{Q_t(a) - \mu_t}{\sigma_t}, \quad (6)$$

where μ_t and σ_t are the mean and standard deviation of $\{Q_t(a')\}_{a' \in \mathcal{A}_t}$. The advantage $A_t(a)$ measures whether the action a performs better than other available intervention actions for the same trajectory and evidence condition. During policy training, the agent stores recent interactions in a buffer and updates the rollback policy periodically every T_π rounds. When candidate outcomes are evaluated for policy learning, we store $\mathcal{D}_t = (o_t, \mathcal{A}_t, \{Q_t(a), A_t(a)\}_{a \in \mathcal{A}_t})$ in a recent interaction buffer $\mathcal{B}_t$. For each buffered interaction, we define the action-group loss:

$$\ell_\pi(o_t, \mathcal{A}_t) = -\frac{1}{|\mathcal{A}_t|} \sum_{a \in \mathcal{A}_t} A_t(a) \log \pi_\theta(a | o_t). \quad (7)$$

To prevent overly large policy shifts from noisy group-relative advantages and stabilize learning, we introduce a KL-regularized loss:

$$\ell_{\text{KL}}(o_t) = \text{KL}(\pi_\theta(\cdot | o_t) \parallel \pi_{\theta_{\text{ref}}}(\cdot | o_t)) \quad (8)$$

where $\pi_{\theta_{\text{ref}}}$ is the reference policy snapshot before the current update. Thus, the overall policy loss is:

$$\mathcal{L}_\pi(\theta) = \mathbb{E}_{(o_t, \mathcal{A}_t) \sim \mathcal{B}_t} [\ell_\pi(o_t, \mathcal{A}_t) + \beta \ell_{\text{KL}}(o_t)]. \quad (9)$$

3.4 Local Trajectory Repair

After the rollback policy selects an intervention action, the local repair module executes the selected action by preserving the reliable prefix and regenerating the potentially unreliable suffix. If the selected action is `continue`, no repair is performed, and the original trajectory is returned, i.e., $\tilde{\tau}_t = \tau_t$. If the selected action is $a_t = \text{rollback_to } k$, the trajectory is decomposed into a preserved prefix $\tau_t^{<k} = \{y_t^{(1)}, \dots, y_t^{(k-1)}\}$ and a repairable suffix $\tau_t^{\geq k} = \{y_t^{(k)}, \dots, y_t^{(K_t)}\}$. $\tau_t^{<k}$ is kept fixed, while the suffix is discarded and regenerated under δ_t :

$$\tilde{\tau}_t^{\geq k} = \text{Regen}(x_t, u_t, H_t, \tau_t^{<k}, \delta_t), \quad (10)$$

where $\text{Regen}(\cdot)$ denotes the reference-conditioned suffix regeneration with the frozen trajectory generator, using δ_t as the evidence reference. The repaired trajectory is then formed as

$$\tilde{\tau}_t = \tau_t^{<k} \oplus \tilde{\tau}_t^{\geq k}. \quad (11)$$

During regeneration, δ_t specifies the repair constraints: the new suffix should follow the current expectation reflected in x_t and u_t , while avoiding stale historical evidence. After repair, the agent returns the final response based on $\tilde{\tau}_t$ and records the repaired trajectory with its repair metadata in H_{t+1} for future inspection.

4 Evaluation

4.1 Experimental Setup

Datasets We evaluate on three benchmarks that cover complementary forms of long-term personalization. **LaMP** (Salemi et al., 2024) evaluates language-model personalization from historical user profiles or interactions and produces responses aligned with individual preferences and patterns. **LongMemEval** (Wu et al., 2025a) focuses on long-term memory retrieval and updating in extended interactions. **LOCOMO** (Maharana et al., 2024) tests conversational memory over evolving user states, past dialogues, and current requests without relying on obsolete or conflicting memories. To expose personalization inertia, we construct preference-conflict interactions following controlled perturbation and contrast-set evaluation (Ribeiro et al., 2020; Gardner et al., 2020).

Models	Methods	LaMP			LongMemEval				LOCOMO			
		ROUGE-L	BERT	Judge	EM	F1	BERT	Judge	EM	F1	BERT	Judge
Qwen3.5-27B	VANILLA RAG	23.1	0.847	57.8	36.8	51.4	0.832	58.6	20.8	33.5	0.806	47.2
	REFLEXION	24.6	0.855	62.3	38.2	52.9	0.839	60.8	24.6	37.1	0.819	52.6
	SELF-REFINE	25.8	0.862	64.9	39.4	54.6	0.845	62.7	25.9	39.8	0.828	56.8
	LD-AGENT	29.4	0.878	71.1	40.5	55.4	0.851	63.6	35.1	47.2	0.855	65.7
	PPP	30.2	0.883	74.6	41.1	55.8	0.857	67.1	33.4	45.1	0.849	68.8
	PIVOT(Ours)	34.8	0.905	82.1	46.7	61.8	0.879	74.2	44.8	57.6	0.883	79.3
LLaMA-3.1-8B	VANILLA RAG	20.7	0.832	50.9	32.4	46.8	0.816	53.1	18.3	30.6	0.794	43.8
	REFLEXION	22.1	0.838	56.1	34.3	48.4	0.824	56.9	21.0	33.8	0.805	49.6
	SELF-REFINE	22.9	0.844	58.7	33.8	49.2	0.827	57.5	22.7	35.4	0.811	52.3
	LD-AGENT	26.5	0.861	65.9	36.4	51.3	0.835	59.8	31.6	42.9	0.842	61.7
	PPP	25.7	0.857	67.4	37.2	51.9	0.838	62.9	29.8	40.8	0.833	64.4
	PIVOT(Ours)	30.6	0.877	76.1	42.0	57.8	0.858	70.4	39.8	52.1	0.865	73.8

Table 1: The overall evaluation results of **PIVOT** and other baseline methods on three benchmark datasets. The best-performing method is marked by **bold**.

For each trajectory, we identify historical preference or memory signals and inject semantically plausible but conflicting current preference cues into multiple later contexts or requests, while keeping the history unchanged before generation.

Baselines We compare our method against state-of-the-art baselines. **Vanilla RAG** (Lewis et al., 2020) retrieves relevant historical memories or user records and directly conditions generation on retrieved contexts. **Reflexion** (Shinn et al., 2023) improves agent behavior through verbal self-reflection and feedback-derived memory. **Self-Refine** (Madaan et al., 2023) iteratively revises the initial response using self-generated feedback. **LD-Agent** (Li et al., 2025) is a long-term personalized agent baseline that maintains user-specific memory for response generation. **PPP** (Sun et al., 2025) represents a personalized preference modeling baseline that incorporates user preference information to improve personalized alignment.

Evaluation Metrics We adopt two complementary categories of metrics for evaluation. For **Task-Level Performance**, we follow the standard evaluation protocol, including ROUGE-L for LaMP; EM and F1 for LongMemEval and LOCOMO, and BERTScore and judge-based evaluation across benchmarks. For **Personalization Inertia**, since aligned responses do not necessarily imply proper handling of evolving user preferences, we additionally evaluate inertia at the trajectory and memory levels, and compute the following metrics: (i) *Preference Conflict Resolution* (PCR), the proportion of cases where current–historical preference conflicts are resolved; (ii) *Stale Evidence Reliance Rate* (SER), the fraction of trajectory steps or used evidence relying on stale evidence; (iii) *Interven-*

tion Precision (IP), the proportion of necessary interventions; (iv) *Memory Contamination Rate* (MCR), the fraction of memory/profile updates containing stale or incorrect preferences.

Implementation Details We conduct our experiments on Qwen3.5-27B (Team, 2026), LLaMA-3.1-8B (Dubey et al., 2024; Touvron et al., 2023). The backbone LLMs generate both initial trajectories and reference-conditioned suffixes. We use the same preference-conflict construction for training and evaluation splits. The prefix consistency head and rollback policy are trained only on the training split and are frozen during evaluation. We default the number of rollback candidates to $M = 2$, the rollback-drop coefficient to $\eta = 1.0$, the intervention cost weight to $\lambda_c = 0.5$, and the KL regularization weight to $\beta = 1.0$. For policy training, grouped intervention outcomes are computed using the same evaluator, and the rollback policy is updated every $T_\pi = 4$ rounds.

4.2 Main Results

We provide the results on LaMP, LongMemEval, and LOCOMO benchmarks under two backbone models in Table 1. **PIVOT** consistently achieves the best performance, demonstrating the effectiveness of inertia-aware trajectory verification and rollback repair. Compared with response-level refinement methods such as REFLEXION and SELF-REFINE, **PIVOT** delivers more consistent gains, suggesting that repairing inertia-affected trajectory segments is more effective than only revising the final response. The improvements are particularly pronounced on LOCOMO, where long-term user states and evolving contexts are more likely to introduce personalization inertia. Similar trends

Methods	LaMP				LongMemEval				LOCOMO			
	PCR $\uparrow$	SER $\downarrow$	IP $\uparrow$	MCR $\downarrow$	PCR $\uparrow$	SER $\downarrow$	IP $\uparrow$	MCR $\downarrow$	PCR $\uparrow$	SER $\downarrow$	IP $\uparrow$	MCR $\downarrow$
VANILLA RAG	43.2	51.5	–	43.0	39.8	56.9	–	49.1	31.6	63.2	–	54.1
REFLEXION	55.4	43.6	53.2	36.8	53.0	48.1	50.2	41.8	45.9	55.2	47.8	47.6
SELF-REFINE	58.1	41.0	55.0	34.5	55.4	45.3	52.1	39.0	49.5	51.8	50.3	44.9
LD-AGENT	58.7	39.6	–	35.8	54.5	43.4	–	40.1	52.2	47.5	–	41.6
PPP	65.0	34.2	64.6	29.5	61.2	38.5	62.5	32.9	60.1	42.0	65.5	35.2
PIVOT(Ours)	76.1	23.9	79.2	17.6	72.0	26.4	80.5	19.7	75.2	23.5	82.4	17.2

Table 2: Evaluation of personalization inertia mitigation under Qwen3.5-27B.

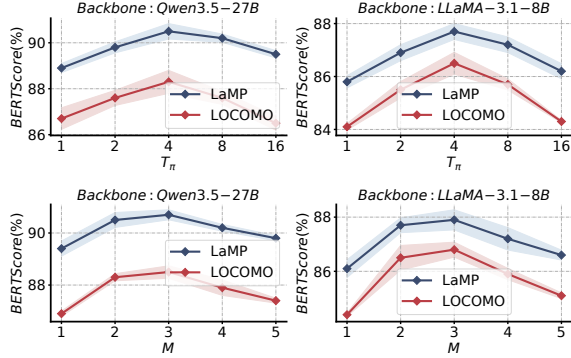


Figure 3: Hyperparameter analysis of T_π and M under different benchmarks and backbones.

are observed on LaMP and LongMemEval, indicating that PIVOT generalizes across both personalized generation and long-memory evaluation settings. Moreover, the gains remain stable across backbone models, showing the generalization of the proposed framework.

Table 2 reports the evaluation of personalization inertia mitigation under Qwen3.5-27B. PIVOT achieves the superior performance across all benchmarks, demonstrating the effectiveness in preserving current preferences, suppressing stale-evidence reliance, and reducing memory contamination. The gains over memory- and profile-based agents indicate that exploiting history alone is insufficient when historical preferences conflict with current expectations. These validations indicate the benefit of prefix-level consistency verification and rollback repair for regulating personalization inertia in long-term personalized agents.

4.3 Hyperparameter Sensitivity Analysis

We conduct hyperparameter analysis of the update interval T_π and candidate size M in Figure 3. The performance consistently follows a rise-then-drop trend as T_π or M increases. When T_π is too small, frequent policy updates may introduce unstable intervention behaviors due to insufficient interaction diversity. In contrast, overly large update intervals weaken the policy’s ability to adapt to evol-

Methods	Avg. Latency	LLM Calls	Extra Tokens	Interv. Rate	Cost/Resolved
VANILLA RAG	5.8	1.0	–	–	–
REFLEXION	9.1	2.3	312	72.4	486
SELF-REFINE	10.5	2.5	428	78.9	512
LD-AGENT	7.9	1.6	–	–	–
PPP	11.4	2.2	216	56.2	398
PIVOT	12.9	2.4	320	38.7	392

Table 3: Latency and repair efficiency on LaMP benchmark under Qwen3.5-27B.

ing personalization dynamics. Similarly, increasing M initially improves performance by providing more candidates for selection, but excessively large candidate sets introduce redundant or low-quality options, which negatively affect decisions.

4.4 Latency and Intervention Efficiency

Table 3 reports the latency and intervention efficiency analysis, including evidence construction for PIVOT. Extra Tokens count online tokens beyond the initial trajectory generation, including evidence construction and suffix regeneration when triggered. Due to online evidence construction, PIVOT does not achieve the lowest latency or LLM call count, but it substantially reduces the intervention rate and achieves the lowest cost per resolved case, suggesting that the additional overhead is converted into more targeted rollback repair.

4.5 Ablation Studies

In Table 4, we report the ablation results on LaMP and LOCOMO. Removing any key component consistently weakens both task performance and inertia mitigation, confirming that the modules of PIVOT are complementary. Specifically, evidence removal causes the largest degradation, confirming its role in exposing conflicts between current expectations and stale history. Removing prefix modeling or consistency drops weakens inertia localization, while removing the rollback policy reduces adaptive intervention. Without regeneration, memory contamination increases, highlighting the necessity of targeted suffix repair.

Methods	LaMP							LOCOMO							
	ROUGE-L	BERT	Judge	PCR	SER	IP	MCR	EM	F1	BERT	Judge	PCR	SER	IP	MCR
w/o Evidence	30.7	0.887	73.6	68.9	31.7	72.4	21.8	37.1	49.8	0.859	68.4	65.2	33.6	75.1	22.7
w/o Prefix	31.4	0.891	75.1	72.3	28.4	74.8	20.6	38.5	51.2	0.864	69.7	68.1	30.9	77.3	21.4
w/o Drop	33.2	0.898	79.4	78.6	22.7	80.2	16.9	42.1	54.9	0.875	75.1	76.3	23.8	84.2	15.8
w/o Policy	32.5	0.895	77.8	76.1	24.8	68.7	18.4	40.9	53.7	0.871	73.2	73.5	26.4	71.6	17.9
w/o Regen	31.9	0.893	76.5	74.8	26.1	77.6	32.6	39.7	52.4	0.868	71.8	71.9	28.2	80.4	35.8
PIVOT	34.8	0.905	82.1	82.7	18.6	84.1	12.7	44.8	57.6	0.883	79.3	81.9	17.8	88.7	11.6

Table 4: Ablation study on the LaMP and LOCOMO benchmarks under Qwen3.5-27B.

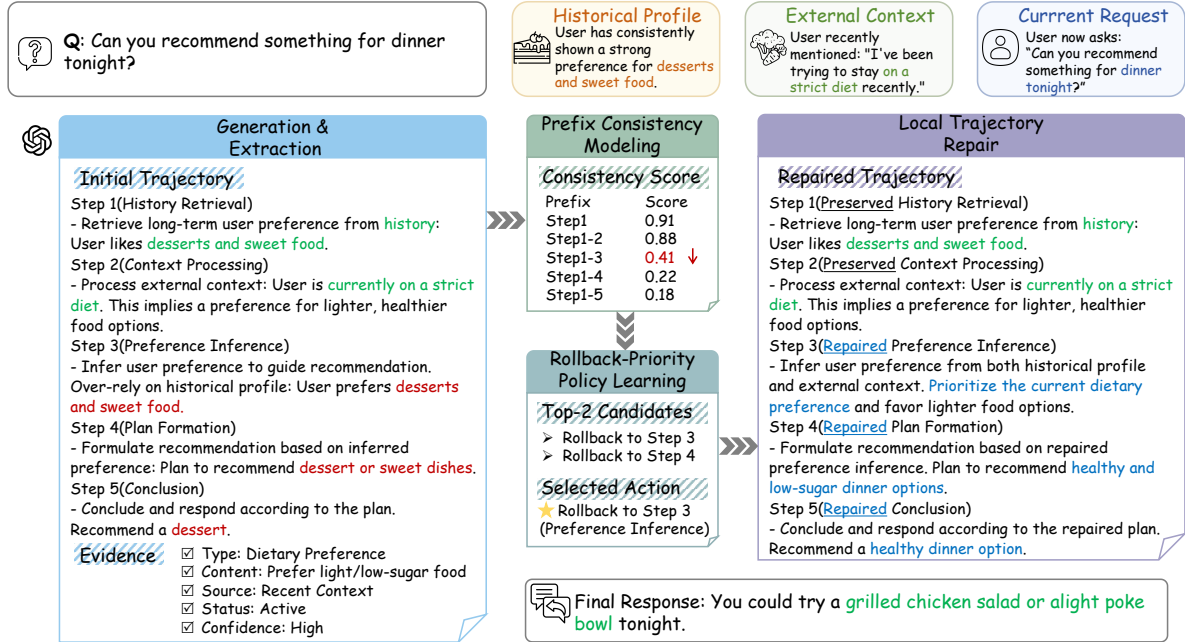


Figure 4: A case study on personalization inertia in long-term LLM agents. The agent initially retrieves a historical preference for **desserts and sweet food**, but later over-relies on it and produces a **diet-incompatible recommendation**. PIVOT constructs trajectory-independent evidence from the current context, **detects a sharp prefix-consistency drop at preference inference**, and selects rollback to the corresponding step. It then preserves the reliable history retrieval and context processing steps while regenerating the affected suffix and yielding a **healthy dinner recommendation**.

4.6 Case Studies and Discussions

We present a representative case study to illustrate how personalization inertia emerges in long-term LLM agents and how PIVOT mitigates such failures. As illustrated in Figure 4, the agent initially retrieves a historical preference for desserts and sweet food, but later over-relies on this stale preference during preference inference and planning, producing a diet-incompatible recommendation despite the current dietary context. During verification, PIVOT constructs trajectory-independent evidence from the current context and localizes the failure through a sharp prefix-consistency drop at the preference inference step. The rollback policy then selects the corresponding semantic step for intervention. Instead of regenerating the entire trajectory, PIVOT preserves reliable history retrieval and context processing while repairing only the

affected suffix under evidence-guided constraints. Thus, the repaired trajectory prioritizes the current dietary preference and produces a healthy dinner recommendation aligned with the latest intent.

5 Conclusion

In this work, we identified personalization inertia as an overlooked failure mode in long-term personalized LLM agents. We proposed PIVOT, an inertia-aware framework that constructs trajectory-independent evidence conditions, verifies prefix-evidence consistency, and performs policy-guided rollback repair before response commitment. By localizing inertia emergence and regenerating only the affected trajectory suffix, PIVOT enables targeted and efficient correction. Experiments on three benchmarks demonstrate the superiority of PIVOT compared with state-of-the-art baselines.

Limitations

This work exhibits several limitations worth noting. First, PIVOT incurs extra computation for evidence construction, prefix scoring, rollback selection, and suffix regeneration. Although it limits decisions to top- M semantic candidates and preserves reliable prefixes, it is still more costly than single-pass generation or standard retrieval-augmented personalization. Second, trajectory-level verification may be unnecessary for simple interactions where historical preferences already align with current expectations.

For future work, we plan to explore adaptive verification policies that condition intervention depth on conflict uncertainty, trajectory complexity, or estimated inertia risk, enabling personalized agents to better trade off reliability and efficiency.

Acknowledgments

This work was supported by the UGC General Research Fund no. 17209822 and the Innovation and Technology Commission Fund no. ITS/383/23FP from Hong Kong.

References

- Xinyun Chen, Maxwell Lin, Nathanael Schaerli, and Denny Zhou. 2024. [Teaching large language models to self-debug](#). In *International Conference on Learning Representations*, volume 2024, pages 8746–8825.
- Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. [Mem0: Building production-ready ai agents with scalable long-term memory](#). *arXiv preprint arXiv:2504.19413*.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, A. Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony S. Hartshorn, Aobo Yang, Archi Mitra, A. Sravankumar, A. Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 510 others. 2024. [The llama 3 herd of models](#).
- Ge Gao, Alexey Taymanov, Eduardo Salinas, Paul Mineiro, and Dipendra Misra. 2024. [Aligning llm agents by learning latent preference from user edits](#). *Advances in neural information processing systems*, 37:136873–136896.
- Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, and 1 others. 2020. [Evaluating models local decision boundaries via contrast sets](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323.
- Zhibin Gou, Zhihong Shao, Yeyun Gong, yelong shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. [Critic: Large language models can self-correct with tool-interactive critiquing](#). In *International Conference on Learning Representations*, volume 2024, pages 57734–57811.
- Yupu Hao, Pengfei Cao, Zhuoran Jin, Huanxuan Liao, Yubo Chen, Kang Liu, and Jun Zhao. 2025. [Evaluating personalized tool-augmented llms from the perspectives of personalization and proactivity](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 21897–21935.
- Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Gong, and Lichao Sun. 2024. [Metatool benchmark for large language models: Deciding whether to use tools and which to use](#). In *International Conference on Learning Representations*, volume 2024, pages 42978–43007.
- Bowen Jiang, Zhuoqun Hao, Young Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo Jose Taylor, and Dan Roth. 2025. [Know me, respond to me: Benchmarking LLMs for dynamic user profiling and personalized responses at scale](#). In *Second Conference on Language Modeling*.
- Jaehyung Kim and Yiming Yang. 2025. [Few-shot personalization of llms with mis-aligned responses](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11943–11974.
- Hannah R Kirk, Alexander Whitefield, Paul Röttger, Andrew Bean, Katerina Margatina, Juan Ciro, Rafael Mosquera, Max Bartolo, Adina Williams, He He, and 1 others. 2024. [The prism alignment dataset: What participatory, representative and individualised human feedback reveals about the subjective and multicultural alignment of large language models](#). *Advances in Neural Information Processing Systems*, 37:105236–105344.
- Ishita Kumar, Snigdha Viswanathan, Sushrita Yerra, Alireza Salemi, Ryan A Rossi, Franck Dernoncourt, Hanieh Deilamsalehy, Xiang Chen, Ruiyi Zhang, Shubham Agarwal, and 1 others. 2024. [Longlamp: A benchmark for personalized long-form text generation](#). *arXiv preprint arXiv:2407.11016*.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. [Retrieval-augmented generation for knowledge-intensive nlp tasks](#). *Advances in neural information processing systems*, 33:9459–9474.
- Hao Li, Chenghao Yang, An Zhang, Yang Deng, Xiang Wang, and Tat-Seng Chua. 2025. [Hello again! llm-powered personalized agent for long-term dialogue](#). In *Proceedings of the 2025 Conference of*

- the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5259–5276.
- Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023. [Making language models better reasoners with step-aware verifier](#). In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 5315–5333.
- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. [Let's verify step by step](#). In *International Conference on Learning Representations*, volume 2024, pages 39578–39601.
- Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. [Faithful chain-of-thought reasoning](#). In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 305–329.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. [Self-refine: Iterative refinement with self-feedback](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 46534–46594. Curran Associates, Inc.
- Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. [Evaluating very long-term conversational memory of llm agents](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13851–13870.
- Charles Packer, Vivian Fang, Shishir_G Patil, Kevin Lin, Sarah Wooders, and Joseph_E Gonzalez. 2023. [Memgpt: towards llms as operating systems](#).
- Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Jianfeng Gao. 2025. [Secom: On memory construction and retrieval for personalized conversational agents](#). In *The Thirteenth International Conference on Learning Representations*.
- Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. 2024. [Personalizing reinforcement learning from human feedback with variational preference learning](#). *Advances in Neural Information Processing Systems*, 37:52516–52544.
- Yilun Qiu, Xiaoyan Zhao, Yang Zhang, Yimeng Bai, Wenjie Wang, Hong Cheng, Fuli Feng, and Tat-Seng Chua. 2025. [Measuring what makes you unique: Difference-aware user modeling for enhancing llm personalization](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 21258–21277.
- Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. [Beyond accuracy: Behavioral testing of nlp models with checklist](#). In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 4902–4912.
- Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2024. [Lamp: When large language models meet personalization](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7370–7392.
- Yu Shang, Peijie Liu, Yuwei Yan, Zijing Wu, Leheng Sheng, Yuanqing Yu, Chumeng Jiang, An Zhang, Fengli Xu, Yu Wang, and 1 others. 2025. [Agentrecbench: Benchmarking llm agent-based personalized recommender systems](#). *Advances in Neural Information Processing Systems*, 38.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. [Reflexion: Language agents with verbal reinforcement learning](#). *Advances in neural information processing systems*, 36:8634–8652.
- Weiwei Sun, Xuhui Zhou, Weihua Du, Xingyao Wang, Sean Welleck, Graham Neubig, Maarten Sap, and Yiming Yang. 2025. Training proactive and personalized llm agents. *arXiv preprint arXiv:2511.02208*.
- Zhen Tan, Jun Yan, I-Hung Hsu, Rujun Han, Zifeng Wang, Long Le, Yiwen Song, Yanfei Chen, Hamid Palangi, George Lee, and 1 others. 2025. [In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8416–8439.
- Qwen Team. 2026. [Qwen3.5: Accelerating productivity with native multimodal agents](#).
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Rodriguez Aurelien, Joulin Armand, Grave Edouard, and Lample Guillaume. 2023. [Llama: Open and efficient foundation language models](#). *arXiv preprint arXiv:2302.13971*.
- Danqing Wang, Kevin Yang, Hanlin Zhu, Xiaomeng Yang, Andrew Cohen, Lei Li, and Yuandong Tian. 2024. [Learning personalized alignment for evaluating open-ended text generation](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13274–13292.

- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. [Self-consistency improves chain of thought reasoning in language models](#). In *The Eleventh International Conference on Learning Representations*.
- Bin Wu, Zhengyan Shi, Hossein A Rahmani, Varsha Ramineni, and Emine Yilmaz. 2024. [Understanding the role of user profile in the personalization of large language models](#). *arXiv preprint arXiv:2406.17803*.
- Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. 2025a. [Long-memeval: Benchmarking chat assistants on long-term interactive memory](#). In *The Thirteenth International Conference on Learning Representations*.
- Shujin Wu, Yi R Fung, Cheng Qian, Jeonghwan Kim, Dilek Hakkani-Tur, and Heng Ji. 2025b. [Aligning llms with individual preferences via interaction](#). In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 7648–7662.
- Yue Xu, Qian Chen, Zizhan Ma, Dongrui Liu, Wenxuan Wang, Xiting Wang, Li Xiong, and Wenjie Wang. 2026. [Toward personalized llm-powered agents: Foundations, evaluation, and future directions](#). *arXiv preprint arXiv:2602.22680*.
- Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z Pan, and 1 others. 2025. [Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning](#). *arXiv preprint arXiv:2508.19828*.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. [Tree of thoughts: Deliberate problem solving with large language models](#). *Advances in neural information processing systems*, 36:11809–11822.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023b. [React: Synergizing reasoning and acting in language models](#). In *The Eleventh International Conference on Learning Representations*.
- Ruifeng Yuan, Shichao Sun, Yongqi Li, Zili Wang, Ziqiang Cao, and Wenjie Li. 2025. [Personalized large language model assistant with evolving conditional memory](#). In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3764–3777.
- Wenhao Yuan, Chenchen Lin, Jian Chen, Jinfeng Xu, Xuehe Wang, and Edith Cheuk Han Ngai. 2026. [Verify before you commit: Towards faithful reasoning in llm agents via self-auditing](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, San Diego, California, United States. Association for Computational Linguistics.
- 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. 2025a. [Personaagent: When large language model agents meet personalization at test time](#). In *First Workshop on Multi-Turn Interactions in Large Language Models (NeurIPS 2025)*.
- Zhehao Zhang, Ryan A. Rossi, Branislav Kveton, Yijia Shao, Diyi Yang, Hamed Zamani, Franck Dernoncourt, Joe Barrow, Tong Yu, Sungchul Kim, Ruiyi Zhang, Jiuxiang Gu, Tyler Derr, Hongjie Chen, Junda Wu, Xiang Chen, Zichao Wang, Subrata Mitra, Nedim Lipka, and 2 others. 2025b. [Personalization of large language models: A survey](#). *Transactions on Machine Learning Research*. Survey Certification.
- Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. 2025a. [Do LLMs recognize your preferences? evaluating personalized preference following in LLMs](#). In *The Thirteenth International Conference on Learning Representations*.
- Weixiang Zhao, Xingyu Sui, Yulin Hu, Jiahe Guo, Haixiao Liu, Biye Li, Yanyan Zhao, Bing Qin, and Ting Liu. 2025b. [Teaching language models to evolve with users: Dynamic profile modeling for personalized alignment](#). *Advances in Neural Information Processing Systems*, 38:33162–33199.
- Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. [Memorybank: Enhancing large language models with long-term memory](#). In *Proceedings of the AAAI conference on artificial intelligence*, pages 19724–19731.

A Evidence Condition Construction

This section describes how we instantiate the evidence constructor $\delta_t = \text{Evi}(x_t, u_t, H_t)$. The goal of δ_t is to provide a trajectory-independent reference that specifies which personalization signals are valid, overridden, contradicted, or uncertain for the current interaction. The constructor only accesses the external context x_t , the current user request u_t , and the interaction history H_t . It does not access the generated trajectory τ_t , the repaired trajectory $\tilde{\tau}_t$, or the final response.

A.1 Evidence Record Format

The output of the evidence constructor is a set of structured records $\delta_t = \{d_{t,j}\}_{j=1}^{N_t}$, where each record is defined as $d_{t,j} = (\text{type}_j, \text{content}_j, \text{source}_j, \text{status}_j, \text{conf}_j)$. The field type_j describes the evidence category, such as `current_constraint`, `current_preference`, `historical_preference`, `explicit_correction`, `task_requirement`, or `profile_summary`. The field content_j stores a normalized natural-language statement of the evidence. The field source_j records the origin of the evidence, such as `current_request`, `external_context`, `recent_feedback`, `episodic_memory`, or `long_term_profile`. The field status_j records the provisional validity of the evidence under the current interaction, and $\text{conf}_j \in [0, 1]$ denotes the confidence of this status assignment. We use four status labels: `active`, `overridden`, `contradicted`, and `uncertain`. An `active` record is currently applicable and should be treated as a valid personalization signal. An `overridden` record is a historical signal superseded by more recent or higher-priority evidence for the current interaction. A `contradicted` record is explicitly incompatible with the current request or context. An `uncertain` record is relevant but cannot be reliably classified as valid or invalid due to ambiguity, weak evidence, or partial conflict.

A.2 Evidence Extraction

The constructor extracts candidate evidence from both the current interaction and the interaction history. From (x_t, u_t) , it extracts current task requirements, explicit instructions, contextual constraints, and newly expressed preferences. From H_t , it extracts previously stored preferences, user-profile

summaries, past corrections, feedback records, and retrieved memories that may influence the current response. These extracted items form the candidate set $\mathcal{R}_t = \mathcal{R}_t^{\text{cur}} \cup \mathcal{R}_t^{\text{hist}}$, where $\mathcal{R}_t^{\text{cur}}$ and $\mathcal{R}_t^{\text{hist}}$ contain current and historical evidence.

Each candidate item is normalized into an evidence statement with a source tag. During construction, we also associate candidate items with auxiliary preference dimensions when possible. These dimensions are used to compare semantically related records and are not included as fields in the final evidence record. A preference dimension denotes the aspect of personalization controlled by the evidence, such as language, tone, response length, formatting style, domain preference, recommendation criterion, safety constraint, privacy requirement, or task-specific objective. For example, “the user usually prefers concise answers” and “please explain this in detail” are compared under the response-length dimension, while “the user prefers English responses” and “answer in Chinese” are compared under the language dimension.

We use an LLM-based structured parser because personalization evidence is often implicit, paraphrased, or context-dependent. The parser is prompted to extract only evidence grounded in (x_t, u_t, H_t) and to return records in the format defined above. It is not given the generated trajectory τ_t , the repaired trajectory $\tilde{\tau}_t$, or the final response. We further apply deterministic post-processing to validate the output schema, normalize labels, remove duplicate records, clip confidence scores to $[0, 1]$, and ensure that each record uses a valid source, status, and confidence value. These checks do not introduce new evidence; they only standardize the parser output. The same extraction prompt and validation procedure are used across all PIVOT variants and ablation settings.

Evidence Condition Construction

You extract structured personalization evidence from the current interaction and user history.

Given the current external context x_t , the current user request u_t , and the interaction history H_t , construct a trajectory-independent evidence condition δ_t . The evidence condition should identify current constraints and preferences, relevant histor-

ical preferences, explicit corrections, and conflicts between current and historical signals.

Current External Context x_t :

{external context, observations, retrieved documents, tool outputs, or environmental constraints available before trajectory generation}

Current User Request u_t :

{current user request}

Interaction History H_t :

{historical interactions, stored memories, user-profile summaries, previous feedback, corrections, or retrieved memory records}

Priority Rule:

When evidence conflicts, use the following priority order from highest to lowest:

1. current explicit request,
2. current external context,
3. recent user feedback,
4. episodic memory,
5. long-term profile or inferred preference.

Status Labels:

- **active:** currently applicable evidence.
- **overridden:** historical evidence superseded by higher-priority current evidence.
- **contradicted:** evidence explicitly incompatible with the current request or context.
- **uncertain:** relevant evidence whose current validity is ambiguous.

Output Requirements:

- Extract only evidence grounded in (x_t, u_t, H_t) .
- Include both current evidence and relevant historical evidence.
- Compare semantically related current and historical evidence when assigning statuses.
- Mark historical evidence as overridden or contradicted when it conflicts with higher-priority current evidence.

- Assign confidence according to source reliability, recency, explicitness, and conflict clarity.
- Do not generate, inspect, assume, or refer to any trajectory, repaired trajectory, or final response.

Output Format:

Return a JSON-style list. Each record must contain:

- **type:** current_constraint, current_preference, historical_preference, explicit_correction, task_requirement, profile_summary, or other.
- **content:** a concise evidence statement.
- **source:** current_request, external_context, recent_feedback, episodic_memory, long_term_profile, or other.
- **status:** active, overridden, contradicted, or uncertain.
- **conf:** a confidence score in $[0, 1]$ for the status assignment.

Example Output:

```
{
  "type": "current_constraint",
  "content": "Answer in Chinese.",
  "source": "current_request",
  "status": "active",
  "conf": 0.98
}
```

A.3 Priority, Conflict, and Status Assignment

After extraction, the constructor compares semantically related records using auxiliary preference dimensions, such as language, response length, tone, or format. These dimensions are used only for grouping during construction and are not included as fields in the final evidence record. Within each group, records are ranked by source priority, recency, and explicitness, following the order specified in the prompt: current explicit request, current external context, recent user feedback, episodic memory, and long-term profile or inferred preference. Status labels are assigned by

checking whether lower-priority records can be jointly satisfied with higher-priority records. If a record is applicable to the current interaction and compatible with higher-priority evidence, it is marked as active. If a historical record is superseded by current evidence but may remain valid in other contexts, it is marked as overridden. If a record is explicitly incompatible with the current request or context, it is marked as contradicted. If its relevance or compatibility is ambiguous, it is marked as uncertain. For example, if the history says that the user prefers concise answers but the current request asks for a detailed proof, the historical conciseness preference is marked as overridden; if the current request explicitly states that a previous preference is no longer desired, that historical preference is marked as contradicted.

A.4 Confidence Estimation

The confidence score conf_j measures the reliability of the assigned status, not the importance of the evidence itself. The LLM-based parser estimates this score using four criteria specified in the prompt: source reliability, recency, explicitness, and conflict clarity. Source reliability reflects the priority of the evidence source. Recency reflects how close the evidence is to the current interaction. Explicitness reflects whether the evidence is directly stated by the user or inferred from prior behavior. Conflict clarity reflects how clearly the record agrees or conflicts with higher-priority evidence in the same dimension.

For an active record, a high confidence score indicates strong support that the evidence should guide the current response. For an overridden or contradicted record, a high confidence score indicates strong support that the evidence should not control the response in this interaction. For an uncertain record, the confidence score is usually lower, indicating that downstream modules should avoid treating it as decisive evidence.

A.5 Construction Algorithm

Algorithm 1 summarizes the construction process. The constructor first uses the fixed LLM-based parser to extract and normalize evidence records from (x_t, u_t, H_t) . It then applies deterministic post-processing to validate the output schema, compare semantically related records, resolve status labels according to the priority and conflict rules, and remove invalid or duplicate records.

Algorithm 1 Evidence Condition Construction

Require: External context x_t , user request u_t , interaction history H_t

Ensure: Evidence condition δ_t

- 1: Prompt the evidence parser with (x_t, u_t, H_t)
 - 2: Extract candidate records $\mathcal{R}_t$ with fields type, content, source, status, and conf
 - 3: Validate the output schema and remove malformed records
 - 4: Normalize source and status to predefined label sets
 - 5: Group semantically related records for conflict checking
 - 6: **for** each group of related records **do**
 - 7: Rank records by source priority, recency, and explicitness
 - 8: Check whether lower-priority records conflict with higher-priority records
 - 9: Update inconsistent status labels according to the priority and conflict rules
 - 10: **end for**
 - 11: Clip confidence scores to $[0, 1]$ and remove duplicate records
 - 12: Form $\delta_t = \{d_{t,j}\}_{j=1}^{N_t}$ from the validated records
 - 13: **return** δ_t
-

A.6 Leakage Prevention

The construction of δ_t follows a strict leakage-prevention protocol. The evidence condition is constructed before prefix consistency estimation, rollback selection, suffix regeneration, or final response generation. The constructor only receives (x_t, u_t, H_t) and never receives the generated trajectory τ_t , the repaired trajectory $\tilde{\tau}_t$, or the final response. The same δ_t is then reused for all downstream modules associated with the current interaction. This protocol prevents δ_t from becoming an oracle signal. If evidence construction were performed after inspecting the trajectory or final response, it could implicitly encode where the agent failed. In contrast, PIVOT detects personalization inertia by comparing generated prefixes against a trajectory-independent evidence reference.

B Prefix Consistency Training

This section describes how we train the prefix-evidence consistency model in § 3.2. Training is performed offline. For each training interaction, we first construct δ_t using the evidence constructor

in Appendix A. Since explicit prefix-level annotations are usually unavailable, we derive weak supervision from the status-annotated evidence condition δ_t and the evidence dependencies in each trajectory step.

Each trajectory step $y_t^{(k)}$ is serialized as $\text{Ser}(y_t^{(k)})$, which records the step type, step output, used evidence or memory, and preference dependencies. We link a step to an evidence record in δ_t if the step explicitly cites the same memory or preference, or if its content semantically matches the evidence statement. Let $\mathcal{E}_t^{(k)} \subseteq \delta_t$ denote the evidence records linked to step k . For prefix $P_t^{(k)}$, the accumulated evidence set is $\mathcal{E}_t^{(\leq k)} = \bigcup_{i=1}^k \mathcal{E}_t^{(i)}$. This linking procedure uses only the trajectory prefix and the preconstructed evidence condition, without accessing repaired trajectories or final responses. We assign a weak consistency label $z_t^{(k)} \in \{0, 1\}$ to each prefix according to the statuses of its linked evidence records:

$$z_t^{(k)} = \begin{cases} 0, & \exists d \in \mathcal{E}_t^{(\leq k)} \text{ s.t. } \text{status}(d) \in \{\text{overridden, contradicted}\}, \\ 1, & \text{otherwise.} \end{cases} \quad (12)$$

Thus, a prefix is labeled inconsistent once it relies on stale or contradicted personalization evidence. Evidence marked as uncertain does not directly flip the label, but lowers the supervision weight. Since the labels are weak, we weight each training example by the confidence of the linked evidence records. For inconsistent prefixes, the weight is determined by the strongest linked overridden or contradicted record. For consistent prefixes, the weight is determined by the active evidence supporting the prefix. Prefixes with only uncertain or weakly linked evidence receive lower weights. We denote the resulting weight as $w_t^{(k)}$. Given the predicted consistency score $c_t^{(k)}$, the consistency head is trained with confidence-weighted binary cross-entropy:

$$\mathcal{L}_c = -\mathbb{E}_{t,k} [w_t^{(k)} (z_t^{(k)} \log c_t^{(k)} + (1 - z_t^{(k)}) \log(1 - c_t^{(k)}))]. \quad (13)$$

The consistency model is implemented as a lightweight encoder and matching head. At inference time, δ_t is constructed online once per interaction, and the trained consistency model scores each prefix using $P_t^{(k)}$ and δ_t without additional LLM calls, producing $c_t^{(k)}$. We then compute the

local consistency drop $\Delta_t^{(k)} = c_t^{(k-1)} - c_t^{(k)}$ for $k \geq 2$. This drop is used as a diagnostic feature for rollback-priority estimation, rather than as causal proof that step k is the source of the error.

C Rollback Outcome Scoring

This section details the outcome evaluator $G_t(\cdot)$ and intervention cost $C_t(\cdot)$ used to train the rollback policy in § 3.3. The outcome scorer is used only during policy training to compare candidate actions under the same observation o_t ; at inference time, the trained rollback policy directly selects an action without enumerating and judging all repaired candidates.

Given an action a , we obtain a candidate trajectory $\hat{\tau}_t(a)$ by either keeping the original trajectory or repairing it from the selected rollback point. The evidence-grounded evaluator $G_t(\cdot)$ measures whether $\hat{\tau}_t(a)$ follows currently valid personalization evidence while avoiding stale or contradicted evidence. We instantiate it with three components:

$$G_t(\hat{\tau}_t(a), x_t, u_t, \delta_t, H_t) = \alpha_e S_{\text{evi}}(a) + \alpha_f S_{\text{faith}}(a) + \alpha_u S_{\text{util}}(a), \quad (14)$$

where $\alpha_e, \alpha_f, \alpha_u$ are non-negative weights that sum to one. $S_{\text{evi}}(a)$ measures whether the candidate respects active evidence records in δ_t and avoids relying on overridden or contradicted records. $S_{\text{faith}}(a)$ measures whether later trajectory steps are supported by earlier trajectory evidence rather than unsupported preference assumptions. $S_{\text{util}}(a)$ measures whether the candidate still satisfies the current request u_t and preserves task-relevant information from x_t and H_t .

In implementation, $G_t(\cdot)$ is computed by a hybrid evaluator. Rule-based checks are used for explicit constraints, such as language, format, refusal, or fixed output requirements. For semantic constraints that cannot be reliably checked by surface matching, we use an LLM-as-judge. The judge receives $(x_t, u_t, H_t, \delta_t)$ and the candidate trajectory $\hat{\tau}_t(a)$, and scores the candidate according to evidence alignment, trajectory faithfulness, and task utility. It is not told which action produced the candidate, which prevents it from favoring rollback actions by construction.

The intervention cost $C_t(a)$ penalizes unnecessary repair. We set $C_t(a) = 0$ for continue. For $a = \text{rollback_to } k$, the cost increases with the

regenerated suffix length and repair token usage:

$$C_t(a) = \lambda_s \frac{K_t - k + 1}{K_t} + \lambda_{\text{tok}} \frac{\text{Tok}(a)}{\text{Tok}_{\max}}, \quad (15)$$

where $K_t - k + 1$ is the number of regenerated trajectory steps, $\text{Tok}(a)$ is the number of tokens consumed by suffix regeneration, and $\text{Tok}_{\max}$ is a normalization constant. This term encourages the policy to prefer smaller local repairs when multiple actions achieve a similar outcome quality.

After computing $Q_t(a)$ for all actions in the same action set, we use the group-relative advantage defined in § 3.3 to train the rollback policy. This normalization compares actions only within the same interaction, reducing the effect of example difficulty. Since the outcome scorer is used for policy training rather than online action selection, the LLM-as-judge does not introduce additional deployment-time LLM calls.

Outcome Scoring Judge

You evaluate a candidate trajectory for a personalized agent.

Given $(x_t, u_t, H_t, \delta_t, \hat{\tau}_t)$, score the candidate trajectory according to:

- **Evidence alignment:** follows active evidence and avoids overridden or contradicted evidence.
- **Trajectory faithfulness:** later trajectory steps are supported by earlier trajectory evidence.
- **Task utility:** addresses the current request and preserves task-relevant information.

Return three scores in $[0, 1]$: evidence_alignment, trajectory_faithfulness, and task_utility, plus one concise rationale.

Do not assume which action produced the candidate, do not reward rollback itself, and do not use information outside $(x_t, u_t, H_t, \delta_t, \hat{\tau}_t)$.

D Evidence-Guided Suffix Regeneration

This section provides the prompt used by the repair module in § 3.4. When the rollback policy

selects $a_t = \text{rollback_to } k$, the prefix $\tau_t^{<k}$ is preserved and the suffix $\tau_t^{\geq k}$ is discarded. The generator then regenerates a new suffix conditioned on the current request, interaction history, preserved prefix, and evidence condition:

$$\tilde{\tau}_t^{\geq k} = \text{Regen}(x_t, u_t, H_t, \tau_t^{<k}, \delta_t). \quad (16)$$

The regeneration prompt is designed to preserve the reliable prefix while forcing the new suffix to follow active evidence and avoid evidence marked as overridden or contradicted. The generator is not allowed to rewrite the preserved prefix or introduce unsupported personalization assumptions.

Evidence-Guided Suffix Regeneration

You regenerate the suffix of a personalized agent trajectory after rollback.

Given the current external context x_t , current user request u_t , interaction history H_t , preserved trajectory prefix $\tau_t^{<k}$, and evidence condition δ_t , generate a repaired trajectory suffix $\tilde{\tau}_t^{\geq k}$.

Current External Context x_t :

{external context, observations, retrieved documents, tool outputs, or constraints}

Current User Request u_t :

{current user request}

Interaction History H_t :

{interaction history and stored memories}

Preserved Prefix $\tau_t^{<k}$:

{trajectory steps before the rollback point}

Evidence Condition δ_t :

{status-annotated evidence records}

Regeneration Requirements:

- Keep the preserved prefix fixed and do not rewrite or contradict it.
- Continue the trajectory from the rollback point with clearly separated semantic steps.
- Follow evidence marked as active.
- Avoid relying on evidence marked as overridden or contradicted.
- Treat uncertain evidence cautiously and do not use it as a decisive personalization signal.

- Do not introduce personalization assumptions unsupported by $(x_t, u_t, H_t, \delta_t)$.
- Ensure the regenerated suffix addresses the current request u_t .

Output Format:

Return only the regenerated suffix:

- Step k : ...
- Step $k + 1$: ...
- ...

The prompt outputs only the regenerated suffix rather than the full trajectory. The repaired trajectory is then obtained by concatenating the preserved prefix and regenerated suffix, $\tilde{\tau}_t = \tau_t^{<k} \oplus \tilde{\tau}_t^{\geq k}$. This design prevents unnecessary rewriting of reliable trajectory steps and keeps the repair localized to the suffix selected by the rollback policy.

E Detailed Dataset Descriptions

We evaluate PIVOT on three long-term personalization benchmarks spanning personalized recommendation, memory-intensive user modeling, and long-context conversational reasoning.

LaMP. LaMP is a benchmark for evaluating language models under personalized language processing tasks. It provides user-specific historical profiles or interaction records together with current inputs, requiring the model to generate responses that are not only task-relevant but also consistent with the user’s past preferences, writing patterns, or behavioral signals. The benchmark covers multiple personalization scenarios, such as personalized classification, recommendation, and generation. These properties make LaMP suitable for evaluating personalization inertia: historical user signals are useful for adaptation, but they may also conflict with the current request or context, causing the agent to over-follow stale preferences. We use LaMP to test whether PIVOT can detect when historical personalization evidence becomes misaligned with current expectations and repair the trajectory before the final response.

LongMemEval. LongMemEval is a benchmark designed to evaluate long-term memory use in conversational agents. It contains extended user-agent interaction histories and queries that require retrieving, updating, or reasoning over user-specific

memories accumulated across time. The benchmark emphasizes realistic memory challenges, including temporal changes, conflicting memories, forgotten corrections, and dependencies between recent and old interactions. These characteristics align closely with our setting, where personalization failures often arise when agents retrieve outdated memories or propagate obsolete user models into current reasoning. We use LongMemEval to examine whether PIVOT can distinguish currently valid memories from overridden or contradicted historical evidence and prevent stale memory usage from contaminating intermediate trajectories.

LOCOMO. LOCOMO is a long-context conversational memory benchmark for evaluating whether models can understand and use information distributed across extended dialogues. It requires agents to answer user-centered questions by grounding responses in long conversational histories, where relevant evidence may be sparse, temporally distant, or entangled with later updates. Compared with short-context personalization datasets, LOCOMO places stronger pressure on memory selection and temporal consistency: an agent must identify which past statements remain valid, which ones have been updated, and how current context changes the interpretation of historical interactions. We use LOCOMO to evaluate whether PIVOT can maintain trajectory faithfulness under long-context memory conditions and regulate personalization inertia when outdated evidence conflicts with the current interaction.

Overall, these benchmarks cover complementary long-term personalization regimes: (i) *profile-based personalization* (LaMP) tests whether agents can use user-specific history without overfitting to stale preferences; (ii) *long-term memory evaluation* (LongMemEval) stresses memory validity, temporal updates, and conflict handling; and (iii) *long-context conversational reasoning* (LOCOMO) evaluates whether agents can ground personalized decisions in extended dialogue histories. This diversity allows us to evaluate PIVOT across settings where personalization inertia may arise from profile misuse, outdated memory retrieval, or temporally inconsistent conversational evidence.

F Comparison Baselines

We compare our proposed method with retrieval-based personalization, reflection-based agent correction, response-level refinement, long-term per-

sonalized agents, and preference-modeling approaches. All baselines are evaluated under the same backbone and are provided with the same current input and available user history.

- VANILLA RAG (Lewis et al., 2020) serves as a retrieval-augmented generation baseline. Given the current context and user request, it retrieves relevant historical memories, user records, or profile entries from the available interaction history and concatenates them with the current input for response generation. This baseline tests whether standard retrieval of personalization evidence is sufficient for handling long-term user-specific interactions. However, VANILLA RAG does not explicitly verify whether retrieved historical memories remain valid under the current context, nor does it localize where stale memories enter the reasoning trajectory. Thus, outdated but semantically relevant memories may be retrieved and used as controlling evidence.
- REFLEXION (Shinn et al., 2023) is an agent correction framework that improves future behavior through verbal self-reflection. After generating an output or observing feedback, the agent produces a reflective summary of what went wrong and stores it as memory for subsequent attempts. In our setting, REFLEXION is adapted to personalized interactions by allowing the agent to reflect on whether its response satisfies the current user request and retrieved user memories. This baseline captures a common self-improvement strategy for agents, where failures are addressed by adding reflective memory. REFLEXION mainly operates after an output-level failure is observed and does not explicitly inspect prefix-level compatibility between intermediate trajectory steps and current personalization evidence.
- SELF-REFINE (Madaan et al., 2023) improves an initial response through iterative self-feedback and revision. The model first generates a response, then critiques its own output, and finally produces a refined response conditioned on the feedback. We adapt SELF-REFINE to the personalization setting by asking the model to check whether the response is consistent with the current request and user history before revising it. This baseline represents response-level correction without explicit trajectory rollback. Since refinement is applied to the generated response as a whole, it can fix some surface-level

personalization errors but may fail to identify which intermediate step introduced stale or contradicted personalization evidence.

- LD-AGENT (Li et al., 2025) is a long-term personalized agent baseline that maintains user-specific memories across interactions and uses them to guide future response generation. It represents systems that explicitly model persistent user information and retrieve or update memory during long-term interaction. In our evaluation, LD-AGENT is used to test whether stronger long-term memory management alone can mitigate personalization inertia. Although such agents can improve personalization by maintaining richer user histories, they may still rely on outdated memory entries when current evidence changes, since they do not perform prefix-level evidence verification or local rollback repair.
- PPP (Sun et al., 2025) represents a personalized preference modeling baseline. It incorporates user preference information into generation or alignment so that model outputs better match user-specific expectations. In our experiments, PPP is used as a preference-aware baseline that can exploit historical user signals beyond simple retrieval. This makes it a strong comparison for personalization accuracy. However, preference modeling methods generally optimize response-level alignment or user preference prediction, rather than detecting when historical preferences become invalid during the current trajectory. Therefore, PPP may still suffer from personalization inertia when stale preferences conflict with current user intent.

G Metric Descriptions

We use two complementary categories of metrics: task-level performance metrics, which measure whether the final response solves the benchmark task, and personalization-inertia metrics, which measure whether the agent properly handles evolving or conflicting personalization evidence during trajectory generation and memory update.

G.1 Task-Level Performance Metrics

Task-level metrics follow the standard evaluation protocols of the corresponding benchmarks. These metrics evaluate the quality of the final response, independent of whether the intermediate trajectory used stale or valid evidence.

ROUGE-L. For LaMP, we report ROUGE-L, which measures the longest common subsequence (LCS) overlap between the generated response p and the reference response r . Let $\text{LCS}(p, r)$ denote the length of their longest common subsequence. ROUGE-L precision and recall are:

$$P_{\text{LCS}} = \frac{\text{LCS}(p, r)}{|p|}, R_{\text{LCS}} = \frac{\text{LCS}(p, r)}{|r|}. \quad (17)$$

ROUGE-L is computed as the F-score of P_{LCS} and R_{LCS} . A higher score indicates stronger sequence-level overlap with the target output.

Exact Match and F1. For LongMemEval and LOCOMO, we report Exact Match (EM) and token-level F1. EM measures whether the normalized prediction p exactly matches the normalized gold answer r :

$$\text{EM} = \mathbb{I}[p = r]. \quad (18)$$

F1 is computed from token precision and recall:

$$P = \frac{|T_p \cap T_r|}{|T_p|}, R = \frac{|T_p \cap T_r|}{|T_r|}, F1 = \frac{2PR}{P + R}, \quad (19)$$

where T_p and T_r are the token sets of the prediction and reference. EM rewards exact answer recovery, while F1 provides partial credit for overlapping answer tokens.

BERTScore. We report BERTScore as a semantic similarity metric across benchmarks. BERTScore computes token-level similarity between contextual embeddings of the generated response and the reference response, and reports a precision-, recall-, or F1-style score. Compared with lexical overlap metrics, it is less sensitive to paraphrasing and surface-form variation.

Judge-Based Evaluation. We use judge-based evaluation to assess response quality when lexical metrics are insufficient. For each instance, the judge is given the current input, available user history, evidence condition, reference answer when available, and the anonymized model response. The judge does not know which method produced the response. It assigns scores in $[0, 1]$ along three dimensions: task correctness, personalization consistency, and evidence faithfulness. Task correctness measures whether the response answers the current request. Personalization consistency measures whether the response follows currently valid

user preferences and constraints. Evidence faithfulness measures whether the response avoids unsupported assumptions or stale historical evidence. The final judge score is the average of the three dimensions:

$$S_{\text{judge}} = \frac{1}{3}(S_{\text{task}} + S_{\text{pers}} + S_{\text{faith}}). \quad (20)$$

The same judge model, prompt, and scoring rubric are used for all compared methods.

Response Quality Judge

You evaluate a personalized agent response.

Given the current request, user history, evidence condition, reference answer if available, and a model response, score the response along:

- **Task correctness:** whether the response answers the current request.
- **Personalization consistency:** whether the response follows currently valid user preferences and constraints.
- **Evidence faithfulness:** whether the response avoids unsupported assumptions and stale historical evidence.

Return three scores in $[0, 1]$ and one concise rationale. Do not assume which method produced the response.

G.2 Personalization-Inertia Metrics

Task-level metrics alone cannot determine whether the agent correctly handles evolving user preferences. A response may match the reference while still relying on stale memories during reasoning, or it may update memory with obsolete preference states. We therefore evaluate personalization inertia using metrics defined over the evidence condition δ_t , the generated trajectory τ_t , the repaired trajectory $\tilde{\tau}_t$, the selected intervention action a_t , and the updated memory state H_{t+1} .

Let $\delta_t = \{d_{t,j}\}_{j=1}^{N_t}$ denote the evidence condition for interaction t . We define the active, stale, and uncertain evidence subsets as:

$$\delta_t^+ = \{d_{t,j} \in \delta_t \mid \text{status}_j = \text{active}\}, \quad (21)$$

$$\delta_t^- = \{d_{t,j} \in \delta_t \mid \text{status}_j \in \{\text{overridden}, \text{contradicted}\}\}, \quad (22)$$

$$\delta_t^? = \{d_{t,j} \in \delta_t \mid \text{status}_j = \text{uncertain}\}. \quad (23)$$

For each trajectory step $y_t^{(k)}$, we denote by $\mathcal{E}(y_t^{(k)}) \subseteq \delta_t$ the evidence records linked to the step, using the evidence-linking procedure described in Appendix B. For a trajectory τ , we denote its steps by $\mathcal{Y}(\tau)$.

Preference Conflict Resolution (PCR). Preference Conflict Resolution measures whether current–historical preference conflicts are resolved in favor of current valid evidence. We first identify the set of conflict instances:

$$\mathcal{C} = \{t \mid \delta_t^+ \neq \emptyset, \delta_t^- \neq \emptyset, \text{Rel}(\delta_t^+, \delta_t^-) = 1\}, \quad (24)$$

where $\text{Rel}(\delta_t^+, \delta_t^-) = 1$ indicates that at least one active evidence record and one stale evidence record refer to the same or semantically related preference dimension. For each $t \in \mathcal{C}$, let τ_t^{out} denote the trajectory finally used for response generation, i.e., $\tau_t^{\text{out}} = \tau_t$ if $a_t = \text{continue}$ and $\tau_t^{\text{out}} = \tilde{\tau}_t$ otherwise. We define:

$$\text{Res}(t) = \mathbb{I} \left[\begin{aligned} &\exists y \in \mathcal{Y}(\tau_t^{\text{out}}), \mathcal{E}(y) \cap \delta_t^+ \neq \emptyset \\ &\wedge \nexists y \in \mathcal{Y}(\tau_t^{\text{out}}), \mathcal{E}(y) \cap \delta_t^- \neq \emptyset \end{aligned} \right] \quad (25)$$

Then PCR is:

$$\text{PCR} = \frac{1}{|\mathcal{C}|} \sum_{t \in \mathcal{C}} \text{Res}(t). \quad (26)$$

A higher PCR indicates that the final trajectory follows current evidence rather than stale historical preferences in conflict cases.

Stale Evidence Reliance Rate (SER). Stale Evidence Reliance Rate measures how often the trajectory relies on stale, overridden, or contradicted evidence. For the final trajectory τ_t^{out} , a step is counted as stale-reliant if it links to any evidence record in δ_t^- . SER is computed over all evaluated trajectory steps:

$$\text{SER} = \frac{\sum_t \sum_{y \in \mathcal{Y}(\tau_t^{\text{out}})} \mathbb{I}[\mathcal{E}(y) \cap \delta_t^- \neq \emptyset]}{\sum_t |\mathcal{Y}(\tau_t^{\text{out}})|}. \quad (27)$$

A lower SER indicates that the agent relies less on outdated personalization evidence during intermediate reasoning.

Intervention Precision (IP). Intervention Precision measures whether triggered interventions are necessary. An intervention is triggered when $a_t \neq \text{continue}$. We define an intervention as necessary if the original trajectory τ_t contains stale evidence

reliance or fails to resolve a current–historical conflict. Let $\mathcal{T}_{\text{int}} = \{t \mid a_t \neq \text{continue}\}$ be the set of interactions where the method intervenes. IP is:

$$\text{IP} = \frac{1}{|\mathcal{T}_{\text{int}}|} \sum_{t \in \mathcal{T}_{\text{int}}} \mathbb{I}[\exists y \in \mathcal{Y}(\tau_t), \mathcal{E}(y) \cap \delta_t^- \neq \emptyset] \quad (28)$$

A higher IP means that interventions are concentrated on cases where the original trajectory actually exhibits personalization inertia, rather than being applied unnecessarily.

Memory Contamination Rate (MCR). Memory Contamination Rate measures whether the memory updates after the interaction writes stale or contradicted personalization information into future history. Let $\Delta H_{t+1} = H_{t+1} \setminus H_t$ denote the newly written memory or profile updates after interaction t . We link each memory update $m \in \Delta H_{t+1}$ to evidence records in δ_t using the same semantic evidence-linking procedure. A memory update is contaminated if it stores or reinforces evidence in δ_t^- . MCR is computed as:

$$\text{MCR} = \frac{\sum_t \sum_{m \in \Delta H_{t+1}} \mathbb{I}[\mathcal{E}(m) \cap \delta_t^- \neq \emptyset]}{\sum_t |\Delta H_{t+1}|}. \quad (29)$$

A lower MCR indicates that the method better prevents inertia-driven errors from being written into memory and reused in future interactions. In practice, the evidence-linking functions $\mathcal{E}(y)$ and $\mathcal{E}(m)$ are implemented using the same matching protocol as prefix consistency training: exact memory identifiers are used when available, and semantic matching is used when evidence is paraphrased. All methods are evaluated with the same evidence condition δ_t and linking procedure for fairness.

Interpretation. Together, these metrics evaluate different aspects of personalization inertia. PCR measures whether conflicts are resolved correctly at the response and trajectory level. SER measures whether stale evidence still appears in intermediate reasoning. IP measures whether interventions are targeted rather than excessive. MCR measures whether inertia-driven states contaminate future memory. Therefore, strong performance requires not only high task-level scores but also high PCR, high IP, low SER, and low MCR.

H Preference-Conflict Construction

We construct preference-conflict interactions to evaluate whether agents can follow current user expectations when they conflict with historical personalization evidence. The key design is to keep

historical evidence available in H_t while introducing a semantically plausible current cue that supersedes or contradicts it.

H.1 Construction Procedure

Given an original interaction (x_t, u_t, H_t) , we construct a perturbed interaction (x'_t, u'_t, H_t) through four steps. First, we identify candidate historical personalization signals p_h from H_t , including explicit user preferences, profile summaries, past corrections, or repeated behavioral patterns. Each candidate is assigned to an auxiliary preference dimension, such as language, response length, format, tone, topic preference, recommendation criterion, or memory update. Second, we select one candidate p_h whose dimension can be naturally modified in the current interaction without changing the core task answer. Third, we create a current preference cue p_c on the same or closely related dimension but with an incompatible value or constraint, such that p_h and p_c cannot both be satisfied in the current interaction. Fourth, we inject p_c into the current request u_t or external context x_t , while keeping H_t unchanged. The resulting instance (x'_t, u'_t, H_t) preserves the stale historical signal as a retrievable distractor, while the current cue should take priority.

The injection location depends on the dataset format. If the benchmark instance contains an explicit user request, we append or rewrite a short current instruction in u_t , such as a language, format, or response-length constraint. If the instance contains a dialogue context or external context, we insert the cue as a later user statement or contextual update in x_t . We avoid modifying facts required to solve the task; only the personalization requirement is changed. For example, if H_t indicates that the user usually prefers concise answers, we may modify the current request to ask for a detailed step-by-step explanation.

H.2 Dataset-Specific Construction

For LaMP, we extract historical signals from user profiles or previous interaction records. These signals often reflect writing style, topic preference, rating tendency, or recommendation criteria. We then add a current cue to the input request that changes the expected personalization behavior for the present instance. For example, a profile-level preference for concise review-style responses can be paired with a current request asking for detailed justification. This evaluates whether the model

over-follows the user profile when the current request provides a valid exception.

For LongMemEval, we use memory records as the source of historical signals. We identify memories that encode user preferences, personal facts, or prior corrections, and construct a later interaction where the user gives a new constraint or correction that supersedes the earlier memory. The old memory remains in H_t , so it can still be retrieved. For example, an earlier memory may indicate that the user prefers short summaries, while the current interaction asks for a comprehensive explanation. This tests whether the agent can distinguish stale memory from current evidence.

For LOCOMO, we construct conflicts from temporally ordered dialogue histories. We identify earlier turns that establish a user preference, state, or habit, and later introduce a current cue that changes the applicable preference for the present request. The earlier conflicting dialogue remains in the long context. For example, an earlier turn may establish a casual tone preference, while the current request asks for a formal response for a professional email. This setting evaluates temporal consistency under long-context memory.

H.3 Perturbation Dimensions and Validation

We construct conflicts over common personalization dimensions, including language, response length, format, tone or style, topic preference, recommendation criterion, and memory update or correction. For each perturbation, the historical signal p_h and current cue p_c must satisfy three conditions: they refer to the same or closely related dimension, they cannot both be fully satisfied in the current interaction, and the injected cue does not change the factual answer or make the task unanswerable.

After construction, we validate each perturbed instance using the evidence-construction procedure in Appendix A. An instance is retained only if δ_t contains at least one current evidence record marked as active and at least one related historical record marked as overridden or contradicted. We also check that the original historical signal remains in H_t and that the perturbation does not introduce unrelated conflicts.

I Details of the Ablation Study

To ensure a fair and reproducible comparison in the ablation experiments reported in Table 4, we clarify how the proposed framework operates

Methods	LaMP				LongMemEval				LOCOMO			
	PCR $\uparrow$	SER $\downarrow$	IP $\uparrow$	MCR $\downarrow$	PCR $\uparrow$	SER $\downarrow$	IP $\uparrow$	MCR $\downarrow$	PCR $\uparrow$	SER $\downarrow$	IP $\uparrow$	MCR $\downarrow$
VANILLA RAG	38.6	56.3	–	47.5	35.2	62.1	–	53.4	27.4	68.7	–	58.9
REFLEXION	50.1	48.4	48.8	41.5	48.2	53.4	45.7	46.6	40.6	60.4	43.1	52.2
SELF-REFINE	52.9	45.7	50.6	39.2	50.7	50.8	47.9	43.9	44.1	57.0	45.9	49.5
LD-AGENT	53.4	44.3	–	40.6	49.6	48.7	–	45.1	46.8	52.8	–	46.4
PPP	59.8	38.9	59.1	34.0	56.1	43.8	57.2	37.7	54.5	47.4	60.1	40.3
PIVOT(Ours)	70.5	28.6	73.9	22.4	66.3	31.8	75.1	24.8	69.0	29.3	77.2	22.1

Table 5: Evaluation of personalization inertia mitigation under LLaMA-3.1-8B.

Methods	Avg. Latency	LLM Calls	Extra Tokens	Interv. Rate	Cost/ Resolved
VANILLA RAG	3.1	1.0	–	–	–
REFLEXION	5.2	2.4	348	76.8	528
SELF-REFINE	6.0	2.6	462	82.1	556
LD-AGENT	4.4	1.7	–	–	–
PPP	6.4	2.3	245	61.5	435
PIVOT	7.1	2.4	356	43.2	428

Table 6: Latency and repair efficiency on LaMP benchmark under LLaMA-3.1-8B.

Methods	Avg. Latency	LLM Calls	Extra Tokens	Interv. Rate	Cost/ Resolved
VANILLA RAG	7.4	1.0	–	–	–
REFLEXION	11.8	2.4	386	78.6	552
SELF-REFINE	13.5	2.6	512	84.3	594
LD-AGENT	10.1	1.7	–	–	–
PPP	14.2	2.3	284	63.8	462
PIVOT	15.4	2.4	386	42.5	438

Table 7: Latency and repair efficiency on LOCOMO benchmark under Qwen3.5-27B.

Methods	Avg. Latency	LLM Calls	Extra Tokens	Interv. Rate	Cost/ Resolved
VANILLA RAG	6.8	1.0	–	–	–
REFLEXION	10.7	2.4	364	76.1	532
SELF-REFINE	12.3	2.6	486	82.0	571
LD-AGENT	9.2	1.7	–	–	–
PPP	13.0	2.3	262	61.4	438
PIVOT	14.2	2.4	358	41.0	421

Table 8: Latency and repair efficiency on LongMemEval benchmark under Qwen3.5-27B.

when specific components are removed. Unless otherwise specified, all remaining modules and training settings remain identical to those used in the full model.

- *w/o Evidence.* This variant removes the trajectory-independent evidence condition δ_t . Instead of constructing status-annotated records from (x_t, u_t, H_t) , the agent directly uses the current input and retrieved history. Thus, it cannot explicitly distinguish active, overridden, contradicted, and uncertain evidence.
- *w/o Prefix.* This variant removes prefix-level consistency modeling. Instead of scoring each partial trajectory $P_t^{(k)}$, it evaluates only the complete trajectory or final response after generation. Therefore, it cannot localize where personalization inertia first appears.
- *w/o Drop.* This variant keeps the prefix consistency scores $\{c_t^{(k)}\}_{k=1}^{K_t}$ but removes the local consistency drop $\Delta_t^{(k)}$. Rollback candidates are ranked using only absolute compatibility scores, without considering whether a specific step causes a sharp decrease in consistency.
- *w/o Policy.* This variant removes the learned rollback policy π_θ . Candidate rollback points are still generated from rollback-priority scores, but the final action is selected by a fixed heuristic:

rollback is triggered when the highest priority score exceeds a preset threshold, and the model rolls back to the corresponding step; otherwise, it continues without repair.

- *w/o Regen.* This variant removes evidence-guided suffix regeneration. The model can still detect potential inertia and select an intervention point, but it does not regenerate the suffix under δ_t . Instead, it keeps the original suffix or applies only response-level revision without freezing the reliable prefix.

J Additional Experimental Results

J.1 Inertia Mitigation

Table 5 evaluates personalization inertia mitigation under the LLaMA-3.1-8B backbone. PIVOT consistently outperforms all baselines across

Methods	LaMP							LOCOMO							
	ROUGE-L	BERT	Judge	PCR	SER	IP	MCR	EM	F1	BERT	Judge	PCR	SER	IP	MCR
w/o Evidence	28.9	0.879	68.5	62.4	37.6	66.3	27.5	33.8	46.1	0.848	62.9	58.7	39.2	69.0	28.6
w/o Prefix	29.6	0.883	70.1	65.8	34.1	68.5	26.1	35.1	47.5	0.853	64.4	61.5	36.5	71.2	27.1
w/o Drop	31.2	0.890	74.3	72.1	27.9	74.5	21.9	38.4	50.9	0.865	69.5	69.6	29.7	78.0	21.0
w/o Policy	30.5	0.887	72.6	69.7	30.0	62.5	23.5	37.2	49.8	0.861	67.7	67.0	32.2	65.8	23.2
w/o Regen	30.1	0.885	71.4	68.4	31.5	71.7	38.4	36.0	48.4	0.857	66.1	65.4	34.0	74.0	41.7
PIVOT	32.7	0.897	76.9	76.1	23.9	79.2	17.6	40.7	53.4	0.873	73.8	75.2	23.5	82.4	17.2

Table 9: Ablation study on the LaMP and LOCOMO benchmarks under LLaMA-3.1-8B.

LaMP, LongMemEval, and LOCOMO, showing that the proposed inertia-aware verification and rollback mechanism remains effective with a smaller backbone model. The improvements over retrieval-, reflection-, refinement-, and preference-modeling baselines indicate that simply retrieving history or revising final responses is insufficient for controlling stale personalization signals. PIVOT also achieves lower stale evidence reliance and memory contamination, suggesting that prefix-level verification and evidence-guided repair help prevent outdated preferences from propagating through trajectories and memory updates.

J.2 Latency and Repair Efficiency Analysis

Tables 6-8 report latency and repair efficiency under different backbones and benchmarks. Across these settings, PIVOT incurs additional overhead from online evidence construction and therefore does not achieve the lowest latency. However, it consistently shows a much lower intervention rate than reflection- and refinement-based baselines, indicating that the rollback policy avoids unnecessary repair. Although PIVOT uses extra tokens for evidence construction and suffix regeneration, it achieves the lowest cost per resolved case among repair-based methods, showing that the additional overhead is converted into more targeted interventions.

The results also reflect benchmark and backbone effects. Under LLaMA-3.1-8B, all methods have lower latency due to the smaller backbone, while PIVOT still maintains the lowest intervention rate and repair cost among intervention-based methods. On LongMemEval and LOCOMO, latency and token overhead increase because longer histories and more complex memory conflicts require more evidence processing and repair. Even in these harder settings, PIVOT remains more efficient than iterative refinement methods in terms of intervention selectivity and cost per resolved case.

J.3 Additional Ablation Study

Table 9 reports the ablation results under LLaMA-3.1-8B. The full PIVOT model consistently achieves the best task-level and inertia-related performance on both LaMP and LOCOMO, showing that all components are complementary. Removing the evidence condition leads to the largest overall degradation, confirming that trajectory-independent evidence is essential for distinguishing current preferences from stale historical signals. Removing prefix consistency or the local drop signal weakens inertia localization, resulting in lower conflict resolution and higher stale evidence reliance. Removing the rollback policy mainly hurts intervention precision, indicating that fixed rollback heuristics are less effective than learned intervention control. Finally, removing evidence-guided regeneration substantially increases memory contamination, showing that detecting inertia is insufficient unless the affected suffix is locally repaired before memory updates.