Inside Out: Evolving User-Centric Core Memory Trees for Long-Term Personalized Dialogue Systems

Anonymous ACL submission

Abstract

Existing long-term personalized dialogue systems struggle to reconcile unbounded interaction streams with finite context constraints, often succumbing to memory noise accumulation, reasoning degradation, and persona inconsistency. To address these challenges, this paper proposes **Inside Out**, a framework that utilizes a globally maintained **PersonaTree** as the carrier of long-term user profiling. By constraining the trunk with an initial schema and updating the branches and leaves, PersonaTree enables controllable growth, achieving memory compression while preserving consistency. Moreover, we train a lightweight **MemListener** via reinforcement learning with process-based rewards to produce structured, executable, and interpretable {ADD, UPDATE, DELETE, NO_OP} operations, thereby supporting the dynamic evolution of the personalized tree. During response generation, PersonaTree is directly leveraged to enhance outputs in latency-sensitive scenarios; when users require more details, the agentic mode is triggered to introduce details on-demand under the constraints of the PersonaTree. Experiments show that PersonaTree outperforms full-text concatenation and various personalized memory systems in suppressing contextual noise and maintaining persona consistency. Notably, the small MemListener model achieves memory-operation decision performance comparable to, or even surpassing, powerful reasoning models such as DeepSeek-R1-0528 and Gemini-3-Pro [1].

1 Introduction

> *Core memories shape Riley's personality islands, with each island serving as a unique emblem of her identity.*
>
> —— *"Inside Out"*

With the rapid advancement of large language models (LLM), dialogue-based agents have demonstrated substantial potential in applications such as personal assistants, affective companionship, and long-term question answering (Chhikara et al., 2025; Rasmussen et al., 2025; Li et al., 2025b). However, within personalized dialogue systems aimed at fostering long-term human-machine trust and emotional connection, a fundamental contradiction exists between the finite context window and the unbounded growth of interaction history (Xiao et al., 2024; Liu et al., 2024). As conversational turns continue to accumulate, the traditional single-context paradigm encounters a severe form of context saturation: indiscriminate aggregation of massive historical information not only drives computational costs sharply upward, but also introduces substantial irrelevant noise, markedly degrading the signal-to-noise ratio. More critically, this unstructured accumulation makes it difficult for the model to accurately extract and sustain a user's personal characteristics from lengthy histories, leading to personalization inconsistency over long-term interactions and thereby seriously undermining user experience and the system's long-term usability (Zhong et al., 2024; Salemi et al., 2024).

To address these challenges, existing studies have primarily explored routes such as explicit profile augmentation and vector-based retrieval, yet neither directly confronts the central bottleneck of personalized memory evolution. Profile-based approaches rely on predefined, static attributes; they are not only slow to update but also struggle to capture implicit cues that users reveal over prolonged interactions, including linguistic style, deeper value orientations, and affective preferences, resulting in superficial personalization modeling (Tan and Jiang, 2023). In contrast, memory-augmented agents based on vector retrieval, while introducing external storage, still essentially treat memory as text fragments or simple lists of facts. Such systems lack an intrinsic, trained decision mechanism for determining which information merits long-term

[1] https://anonymous.4open.science/r/PersonaTree

retention, and instead often depend on rigid heuristics or elaborate prompt engineering (Liu et al., 2023). This accumulation of memories without value-based judgment causes the memory repository either to become bloated and uninterpretable due to noise accretion, or to lose the long-range logical thread through fragmentation of key context, ultimately failing to sustain a vivid and coherent persona (Yoran et al., 2024).

This discrepancy between memory accumulation and core persona formation" motivates us to return to the foundations of human cognition for an answer. As illustrated by the film "*Inside Out*", individual identity does not stem from a simplistic stacking of all experiences, but rather is constructed upon core memories that shape distinct "Islands of Personality". This aligns with theoretical findings in cognitive psychology, such as Self-Schema theory (Markus, 1977; Tikka and Oinas-Kukkonen, 2019), which emphasizes that humans maintain a stable self-concept by filtering and hierarchically organizing key memories.

Inspired by these insights, we propose the **Inside Out** framework, which aims to grow an evolvable user core memory tree "from the inside out" through unbounded interactions. Firstly, to delineate the theoretical boundaries of the memory tree, we construct a hierarchical Schema based on the Biopsychosocial model, scientifically decomposing user characteristics into three core dimensions. This interdisciplinary Schema design establishes the initial structure of the user **PersonaTree**. Secondly, to endow the system with dynamic evolution, we propose an iterative tree-update mechanism and introduce a reinforcement learning (RL) strategy based on process rewards to train a lightweight model, **MemListener**. This model learns to compress a continuous stream of unstructured dialogue in real time into standardized tree-structured operations, encoding user core features within the branch and leaf nodes. Finally, addressing the trade-off between efficiency and effectiveness during the inference stage, this paper designs an adaptive response generation mechanism: In latency-sensitive scenarios, a fast mode is enabled to perform reasoning directly based on the PersonaTree. When facing long-tail detail requirements, the system switches to the agentic recall mode, utilizing the PersonaTree to guide deep retrieval. The primary contributions of our work are summarized as:

- We propose PersonaTree, grounded in the biopsychosocial schema. By transforming unstructured dialogue streams into standardized atomic tree operations in real-time, PersonaTree achieves the dynamic compression, explicit management, and high signal-to-noise ratio maintenance of implicit user profiles.
- We design a training strategy utilizing RL with process rewards. Leveraging the constructed dataset of 28k instructions, we train a lightweight model, MemListener, to execute precise memory editing.
- Our experiments reveal the potential of a collaborative paradigm where "small models maintain memory while LLMs handle generation". Results show that MemListener achieves memory-decision performance comparable to strong reasoning models, and that PersonaTree offers a new pathway toward low-cost, highly reliable deployment of long-term personalized dialogue systems.

2 Related Works

2.1 Personalization and Memory

Personalization aims to adapt a dialogue system's linguistic style and interaction policy to a specific user's stable traits and evolving state. In interactive settings, personalization is inherently coupled with memory: models must distill reusable user representations from past interactions and fuse them during generation. Li et al. (2016) proposed persona-based dialogue generation to mitigate inconsistency and lack of personality in open-domain dialogue, and Zhang et al. (2018) formalized the PersonaChat task. Subsequent studies emphasized multi-dimensional user attributes. For example, Zheng et al. (2019) introduced the large-scale multi-turn dataset PersonalDialog. In parallel, Madotto et al. (2019) framed personalization as a meta-learning problem to enable few-shot adaptation. In the LLM era, Chen et al. (2024) systematically reviewed major directions in personalized dialogue generation, while Tan et al. (2024) assigned parameter-efficient personalization modules to users to improve multi-task personalization.

2.2 LLM Agents with External Memory

To overcome the limitations of LLMs' finite context windows and endow them with capabilities for continuous learning and long-term interaction,

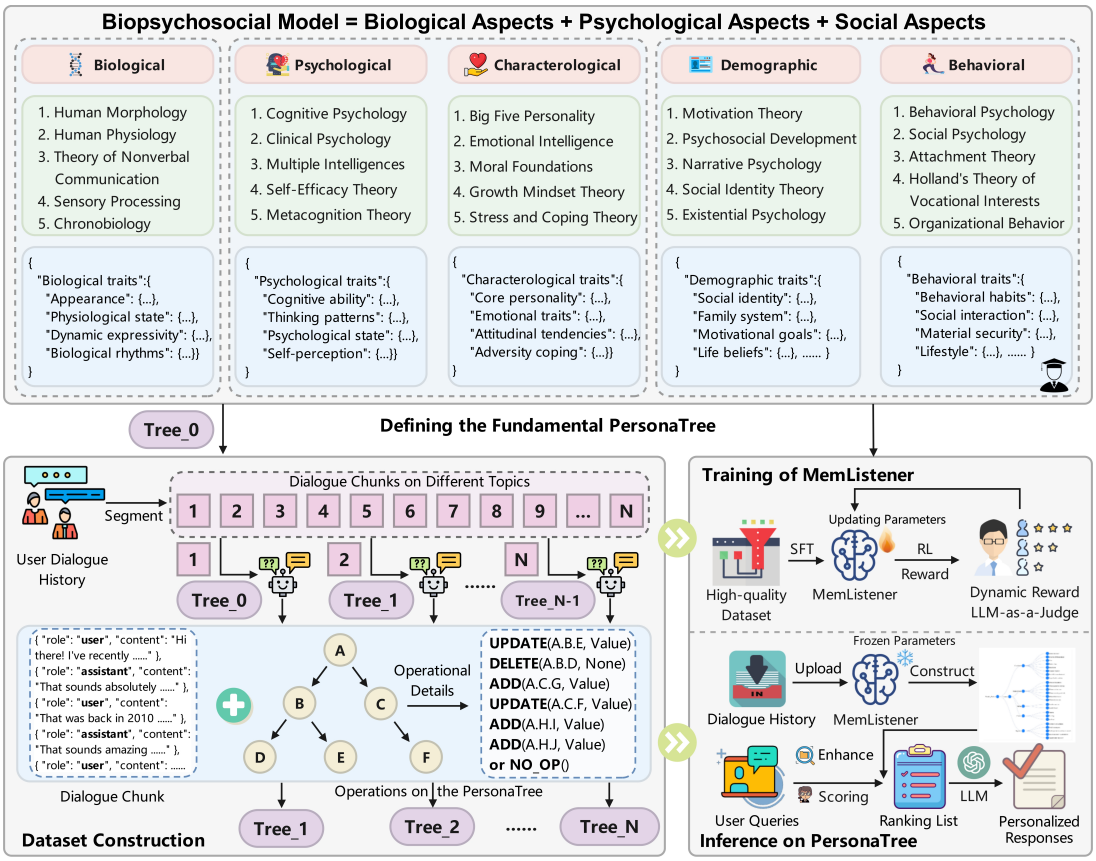


Figure 1: Overview of the entire process of our Inside Out framework.

constructing memory systems has emerged as a pivotal research direction. LangMem [2] enables continual learning and cross-session consistency by decoupling hot-path memory primitives from backend asynchronous integration. Mem0 (Chhikara et al., 2025) adopts a multi-level memory architecture to support multi-session retrieval and personalization at relatively low overhead. A-Mem (Xu et al., 2025) builds an evolvable memory network via self-organizing indexing and linking mechanisms. MemoryOS (Kang et al., 2025) manages short, medium, and long-term memory through OS-style hierarchical storage together with corresponding update and retrieval policies to preserve contextual coherence.

3 The Inside Out Framework

3.1 Overview Architecture

This study proposes the Inside Out Framework, which aims to address the challenges of personalized consistency and contextual forgetting in long-term dialogues through a structured memory evolution mechanism.

[2] https://github.com/langchain-ai/langmem

Framework Pipeline. As shown in Figure 1, the framework consists of three key modules: Dynamic PersonaTree Evolution (Section 3.2), MemListener Training (Section B), and Adaptive Response Generation (Section 3.4). First, **PersonaTree and dataset construction** initializes a persona tree based on the Biopsychosocial Model, segments the user's dialogue history into consecutive dialogue chunks, and generates operations on the PersonaTree, thereby constructing a memory evolution dataset. Second, **MemListener training** leverages the resulting high-quality dataset to update the parameters of MemListener via supervised fine-tuning (SFT) and RL with a dynamic reward mechanism, enabling it to extract structured memory from unstructured dialogues. Finally, **PersonaTree inference** freezes the MemListener parameters at application time, reconstructs the attribute tree from the dialogue history, uses this structured memory to enhance user queries, and ultimately generates personalized responses through an LLM.

Problem Formulation. We define the task of a personalized dialogue system as a process of maximizing response utility over an infinitely long di-

alogue stream. Given a user U with a historical dialogue sequence $H = \{x_1, y_1, ..., x_t, y_t\}$, where x denotes user inputs and y denotes system responses, conventional context-window approaches attempt to directly model $P(y_t \mid H_{t-k:t})$, but are constrained by the window length k. Our framework introduces an explicit, structured user state $\mathcal{T}$ (i.e., PersonaTree), thereby reformulating the problem as state tracking and state-conditioned generation. The goal is to learn a state update function f_{update} such that:

$$\mathcal{T}_t = f_{\text{update}}(\mathcal{T}_{t-1}, D_t) \tag{1}$$

$$y_t = f_{\text{gen}}(x_t, \mathcal{T}_t, f_{\text{recall}}(\mathcal{T}_t, H)) \tag{2}$$

where D_t denotes the current dialogue chunk, f_{gen} produces the system reply given the current user input and the tracked user state, and f_{recall} is a retrieval function that recalls relevant historical snippets from the full dialogue history H conditioned on the current state $\mathcal{T}_t$.

3.2 Dynamic PersonaTree Evolution

PersonaTree Initialization. At system startup, we construct an initial PersonaTree to serve as the starting point of the user's long-term structured state. Specifically, we first determine the set of writable trunk and leaf fields according to a predefined unified schema, and constrain the storage type of each leaf node to a descriptive string, which is used to hold a compressed summary of the user's core personalized attributes. This design ensures that memory capacity remains controllable and prevents unbounded growth as the dialogue progresses. The schema is informed by interdisciplinary human-factors and psychological theory frameworks, with its theoretical grounding illustrated in Figure 1. Subsequently, under the schema constraints, we initialize the leaf nodes (allowing empty strings or default placeholder text), thereby obtaining the initial persona tree $\mathcal{T}_0$. The specific initial PersonaTree instance adopted in this paper is provided in Appendix D.

Iterative PersonaTree Updating. To enable scalable maintenance of long-term personalized memory over an infinitely long dialogue stream, we adopt an iterative updating mechanism: any input modality (historical file import, short-snippet input, or real-time cache triggering) is normalized into a dialogue-chunk sequence $(D_1, \ldots, D_N)$, and for each D_t we execute a closed-loop update of operation-list generation, safe parsing and execution, versioned persistence.

Step 0: System Loading. The system loads the text fields of all leaf nodes, yielding the initial state $\mathcal{T}_0$. Meanwhile, the task specification and system constraints are abstracted into a rule set $\mathcal{R}$, including update rules, writable scope, leaf constraints, and output format.

Step $t = 1, \ldots, N$. For any dialogue chunk D_t, the system executes the following three stages:

(a) State Construction. Given a dialogue chunk D_t, set $\text{Input}_t \leftarrow D_t, \mathcal{T}_{t-1}$.

(b) Operation List Generation. Conditioned on $(D_t, \mathcal{T}_{t-1}, \mathcal{R})$, LLM outputs an operation list $\mathcal{O}_t$, consisting of one or more atomic operations that strictly follow a predefined operation grammar. The operation types are limited to:

- `ADD(path, value)`: write descriptive text to the specified path; if the path does not exist, it may be created under an extended-schema policy;
- `UPDATE(path, value)`: perform an overwrite rewrite on the target leaf node, updating its text to the new value;
- `DELETE(path, value)`: clear the target leaf node or write a deletion marker to indicate that this type of information should be removed from long-term memory;
- `NO_OP()`: the current dialogue chunk does not contain stable core persona information that should be written to the PersonaTree.

For update operations, our framework unifies them as rewrites of leaf strings. More importantly, potential conflicts between new and old information are resolved by LLM during the generation of $\mathcal{O}_t$. Based on D_t and the contextual old values in $\mathcal{T}_{t-1}$, the model must decide whether to overwrite prior information, preserve salient change cues, or solely append new information. In other words, conflict resolution is explicitly lifted to the policy-generation stage, so as to leverage the LLMs' holistic inference over semantics, temporal order, and narrative consistency.

(c) Parsing and Execution. This module serves as a safety gate that enforces structural and capacity constraints when applying $\mathcal{O}_t$: it validates that each path targets a permissible leaf, ensures each value is a string or an allowed deletion marker to

avoid parsing ambiguity and state pollution, and applies length control by compressing any overlong value to satisfy the per-leaf budget. Importantly, it performs no conflict resolution or secondary semantic rewriting; it only executes the prescribed operations under these constraints.

(d) Versioned Persistence. After the execution, the updated tree state $\mathcal{T}_t$ is materialized and persisted as a new version, either serialized to a JSON file or stored in a JSON-capable database (e.g., document stores such as MongoDB). Iterating over $t = 1, \ldots, N$ yields a traceable evolution sequence $\{\mathcal{T}_0, \ldots, \mathcal{T}_N\}$, with $\mathcal{T}_N$ serving as the compressed long-term memory for retrieval-augmented and personalized generation at the final task-query stage.

3.3 MemListener Training

Training Data Synthesis. During training data construction, we select subsets from HaluMem (Chen et al., 2025b) and PersonaMem (Jiang et al., 2025) that are relevant to implicitly characterizing user-specific attributes as the raw corpus sources. Using the dynamic PersonaTree evolution procedure described in Section 3.2 as the backbone, we invoke DeepSeek-R1-0528 to generate supervision signals for training.

Warm-up via SFT. We first perform full supervised fine-tuning to initialize the base model as a MemListener that can stably generate operation lists. For any training sample, let the input context be s (including the dialogue chunk, the previous tree state, and rule constraints), and the target output be o (the ground-truth operation sequence segment). We optimize a standard autoregressive cross-entropy objective:

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\frac{1}{\tau} \sum_{t=1}^{\tau} \log P_\theta(o_t \mid o_{<t}, s)\,,$$

where θ denotes the model parameters and τ is the target sequence length.

Alignment via Process-Reward RL. After the SFT warm-up, we continue alignment with the remaining data using RL driven by process-based rewards. We set the model's maximum context length to 11K tokens, with the input length capped at 10K, which constitutes a typical ultra-long sequence optimization setting. Therefore, we enable the DAPO loss for process-reward alignment within a GRPO framework. We perform credit assignment via token-level policy gradients, and mitigate entropy collapse by decoupling the clipping range from dynamic sampling constraints, ensuring that the within-group advantage retains nonzero variance. This makes the method better suited to long-chain reasoning and structured operation-sequence generation. In addition, by limiting the maximum generation length and applying dynamic resampling to filter degenerate groups, we reduce training noise introduced by truncation and within-group advantage degeneration.

Concretely, for each training sample, we take s (including the dialogue chunk, the previous tree state, and update constraints) as input, and use the manually verified reference output $y^\star$ as the ground-truth operation trajectory. During optimization, we sample a group of candidate outputs $\{y_i\}_{i=1}^{G}$ from the old policy $\pi_{\theta_{\mathrm{old}}}(\cdot \mid s)$, compute sequence-level returns under the process reward function $R(\cdot)$ as $R_i = R(y_i, y^\star; s)$, and then update the policy parameters θ using the DAPO objective:

$$\mathcal{J}_{\mathrm{RL}}(\theta) = \mathbb{E}_{(s,y^\star)\sim\mathcal{D},\ \{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\mathrm{old}}}(\cdot|s)} \Bigg[\frac{1}{\sum_{i=1}^{G}|y_i|} \sum_{i=1}^{G}\sum_{t=1}^{|y_i|} \min\Big(r_{i,t}(\theta)\,\hat{A}_{i,t},\ \mathrm{clip}\big(r_{i,t}(\theta),\ 1-\varepsilon_{\mathrm{low}},\ 1+\varepsilon_{\mathrm{high}}\big)\,\hat{A}_{i,t} \Big) \Bigg],$$

$$\text{s.t. } 0 < \Big| \{\, y_i \mid \mathrm{is\_equivalent}(y^\star, y_i) \,\} \Big| < G.$$

Policy updates are based on the importance ratio

$$r_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t} \mid s, y_{i,<t})}{\pi_{\theta_{\mathrm{old}}}(y_{i,t} \mid s, y_{i,<t})},$$

$$\hat{A}_{i,t} = \frac{R_i - \mathrm{mean}\Big(\{R_j\}_{j=1}^{G}\Big)}{\mathrm{std}\Big(\{R_j\}_{j=1}^{G}\Big)},$$

where $R_i \in [-1, 1]$ denotes the sequence-level score assigned by the evaluator for the i-th sampled output y_i in a group of size $G = 8$. We apply within-group standardization to obtain the advantage estimate $\hat{A}_{i,t}$, thereby improving training stability. We adopt $\varepsilon_{\mathrm{low}} = 0.2$ and $\varepsilon_{\mathrm{high}} = 0.28$ to relax the upper-bound clipping, providing greater update headroom for increasing the probabilities of low-probability exploratory tokens. The detailed data construction, training parameter settings, and the design of the reward function are presented in Appendix B.

3.4 Adaptive Response Generation

During the inference, we treat the final tree state $\mathcal{T}_N$ as structured long-term memory and adopt an adaptive response strategy to satisfy both low-latency and high-coverage requirements.

PersonaTree-Augmented Generation. For the latency-sensitive interaction scenario, the system enables a lightweight fast mode: it directly reads out the structure of $\mathcal{T}_N$ along with the non-empty leaf texts as a personalized prior, concatenates them with the user query q as the input context, and generates an answer in a single pass.

Agentic Recall and Fusion. When the user explicitly requests additional details or the query exhibits stronger long-tail characteristics, the system switches to the agentic recall mode. Concretely, we generate a set of expanded queries $\{\tilde{q}^{(k)}\}_{k=1}^{K}$ from the original query q conditioned on $\mathcal{T}_N$, where each $\tilde{q}^{(k)}$ emphasizes a different attribute dimension or potential missing aspect relevant to the question. The system retrieves candidate evidence sets $\{d_j^{(k)}\}$ in parallel for $\{q^{(k)}\}$, and reranks them based on relevance to obtain a fused context C. Finally, we generate the final answer conditioned on $[q, \mathcal{T}_N, C]$. This procedure operates under a gated policy that is triggered only when necessary, improving answer quality in complex scenarios while keeping the overall interaction cost controllable.

4 Experiments

4.1 Experimental Setup

Datasets and Metrics. We conduct experiments on the PersonaMem benchmark (Jiang et al., 2025). This dataset is centered on user personas: each instance contains a user's static demographic information as well as dynamic attributes that evolve over time. For interaction-history construction, each history consists of approximately 10 multi-turn conversations concatenated in chronological order, resulting in an overall context length of about 32k tokens and covering 15 categories of real-world tasks that require personalization. For evaluation, we use accuracy as the primary metric, reporting overall accuracy and further breaking it down into per-skill accuracies across seven query-skill categories. Under the discriminative setting, the model directly outputs the selected option.

Baselines. To evaluate the effectiveness of the proposed method, we compare against two standard interaction paradigms Only LLM and Full Context as well as four representative memory-management frameworks LangMem, Mem0 (Chhikara et al., 2025), A-Mem (Xu et al., 2025), and MemoryOS (Kang et al., 2025). For each baseline, we use the best-performing configurations reported in the original works (e.g., agentic retrieval). Detailed descriptions and experimental setups are provided in Appendix C.1.

Implementation Details. In our approach, the construction of core training data leverages the DeepSeek-R1-0528. To encourage diverse generations, we set temperature to 0.7 and top-p to 0.9. For retrieval components, both our method and all baseline systems share the same settings: we use BGE-M3 as the retriever and BGE-Reranker-Large as the re-ranking model. We set the retrieval count to 4 and ensure that different methods maintain a fundamentally consistent length of retrieval context. All training and evaluation experiments are conducted on a single node equipped with 8 NVIDIA H200-141GB GPUs.

4.2 Main Results

As shown in Table 1, we conduct a unified evaluation across three response models: DeepSeek-V3.1, Longcat-Flash-Chat, and DeepSeek-R1-0528, while fixing DeepSeek-R1-0528 as the memory extractor for all settings.

On DeepSeek-V3.1, our best configuration achieves $\mathrm{Overall} = 71.31$, improving by $+18.68$ over Only LLM and by $+7.47$ over ALL Dialogue, and further exceeding the strongest comparative memory system, MemoryOS, by $+8.83$. On Longcat-Flash-Chat, PersonaTree-ALL with Qwen3-8B-RL attains the best overall performance with $\mathrm{Overall} = 75.38$, improving by $+13.58$ over ALL Dialogue and by $+10.35$ over MemoryOS. Under this setting, all sub-metrics exhibit consistent improvements, indicating that after dialogue compression and multi-round retrieval fusion, the model can more accurately recover the user's factual background, characterize the temporal evolution of preferences, and generate new content with broader coverage.

When using DeepSeek-R1-0528 as the response model, we observe similarly substantial improvements: PersonaTree-ALL with Qwen3-8B-RL reaches $\mathrm{Overall} = 76.06$, outperforming ALL Dialogue by $+11.20$ and surpassing MemoryOS by $+13.41$. In terms of fine-grained metrics, Pref-Rec

Methods	Overall	Recall-Facts	Pref-Rec	New-Ideas	Recall-Reason	Pref-Evol	Gen-New	Recall-User
DeepSeek-V3.1 (Response)								
Only LLM	52.63	59.69	43.64	5.38	80.81	65.47	42.11	52.94
ALL Dialogue	63.84	78.29	61.82	12.90	81.82	65.47	77.19	76.47
LangMem+DeepSeek-R1-0528	57.05	70.54	43.64	11.83	80.81	58.99	63.16	70.59
Mem0+DeepSeek-R1-0528	60.44	79.84	52.73	5.38	83.84	61.15	64.91	82.35
A-Mem+DeepSeek-R1-0528	59.76	79.84	54.55	4.30	86.87	56.83	64.91	76.47
MemoryOS+DeepSeek-R1-0528	62.48	72.87	**65.45**	11.83	84.85	63.31	73.68	76.47
PersonaTree-ALL+DeepSeek-R1-0528	71.14	88.37	56.36	22.58	88.89	71.22	**89.47**	88.24
PersonaTree-ALL+Qwen2.5-7B-RL	**71.31**	**89.15**	61.82	**23.66**	**91.92**	69.06	85.96	76.47
PersonaTree-ALL+Qwen3-8B-RL	70.80	84.50	61.82	21.51	87.88	72.66	85.96	**100.00**
w/ PersonaTree+Router	70.12	82.17	63.64	19.35	88.89	**73.38**	84.21	94.12
w/ PersonaTree	61.97	75.19	60.00	4.30	83.84	64.75	75.44	88.24
Longcat-Flash-Chat (Response)								
Only LLM	54.33	62.02	43.64	8.60	84.85	71.22	28.07	52.94
ALL Dialogue	61.80	79.07	63.64	15.05	87.88	53.96	68.42	70.59
LangMem+DeepSeek-R1-0528	58.23	75.97	47.27	8.60	82.83	64.03	50.88	64.71
Mem0+DeepSeek-R1-0528	59.59	80.62	52.73	5.38	85.86	62.59	49.12	76.47
A-Mem+DeepSeek-R1-0528	60.95	82.17	54.55	8.60	85.86	62.59	56.14	64.71
MemoryOS+DeepSeek-R1-0528	65.03	77.52	**72.73**	11.83	86.87	66.91	70.18	76.47
PersonaTree-ALL+DeepSeek-R1-0528	72.67	88.37	**72.73**	26.88	92.93	69.78	85.96	64.71
PersonaTree-ALL+Qwen2.5-7B-RL	73.34	**93.02**	67.27	27.96	92.93	67.63	**89.47**	70.59
PersonaTree-ALL+Qwen3-8B-RL	**75.38**	**93.02**	70.91	**30.11**	**93.94**	**71.94**	85.96	**88.24**
w/ PersonaTree+Router	71.82	89.92	70.91	22.58	90.91	68.35	84.21	82.35
w/ PersonaTree	65.20	79.84	67.27	10.75	87.88	68.35	68.42	76.47
DeepSeek-R1-0528 (Response)								
Only LLM	44.14	30.23	38.18	19.35	76.77	55.40	38.60	41.18
ALL Dialogue	64.86	69.77	63.64	11.83	84.85	73.38	84.21	70.59
LangMem+DeepSeek-R1-0528	54.84	63.57	61.82	15.05	78.79	56.83	50.88	41.18
Mem0+DeepSeek-R1-0528	49.41	32.56	56.36	24.73	81.82	56.83	43.86	58.82
A-Mem+DeepSeek-R1-0528	47.37	30.23	45.45	22.58	77.78	59.71	36.84	76.47
MemoryOS+DeepSeek-R1-0528	62.65	62.79	78.18	11.83	77.78	70.50	78.95	**82.35**
PersonaTree-ALL+DeepSeek-R1-0528	74.87	80.62	69.09	27.96	94.95	82.01	**89.47**	**82.35**
PersonaTree-ALL+Qwen2.5-7B-RL	74.70	79.84	80.00	27.96	94.95	82.01	84.21	64.71
PersonaTree-ALL+Qwen3-8B-RL	**76.06**	**80.62**	**81.82**	**29.03**	92.93	**84.17**	87.72	76.47
w/ PersonaTree+Router	74.19	**80.62**	67.27	25.81	**95.96**	83.45	84.21	76.47
w/ PersonaTree	65.70	71.32	65.45	18.28	91.92	69.78	75.44	64.71

Table 1: Main experimental results are presented on three different response models. Recall-Facts, Pref-Rec, New-Ideas, Recall-Reason, Pref-Evol, Gen-New, and Recall-User respectively denote recalling user-shared facts, providing preference-aligned recommendations, suggesting new ideas, recalling reasons behind preference updates, tracking preference evolution, generalizing to new scenarios, and recalling user-mentioned facts. **Bold** and underlined numbers denote the best and second-best results, respectively.

and New-Ideas improve by +18.18 and +17.20 over ALL Dialogue, respectively, indicating that structured memory combined with process-aligned generation can substantially strengthen preference consistency and creative completion. Meanwhile, the method maintains stable advantages on Recall-Reason and Pref-Evo, suggesting that PersonaTree offers greater interpretability and controllability in preserving and invoking causal chains and evolution cues. More comprehensive comparative experiments, ablation analyses and visual presentations are shown in Appendix C.

Methods	Evaluation Models			Avg. Length of Context
	DS-V3.1	Longcat	DS-R1-0528	
Only LLM	52.63	54.33	44.14	0
ALL Dialogue	**63.84**	61.80	64.86	32K
PersonaTree				
+Qwen2.5-7B-Instruct	55.18	55.18	50.08	1852.08
+Qwen3-8B	52.97	54.33	47.71	1392.49
+GPT-4o-mini	55.86	58.23	55.18	1154.35
+Longcat-Flash-Chat	58.91	61.63	60.78	2305.46
+DeepSeek-V3.1	60.03	62.31	61.80	2227.78
+DeepSeek-V3.2	60.32	63.50	63.16	2292.54
+DeepSeek-R1-0528	60.61	63.33	63.50	1844.19
+Gemini-3-Pro	61.29	63.16	63.16	2252.89
+Qwen2.5-7B-Instruct				
+SFT	60.27	62.82	60.61	2158.03
+SFT+RL	62.82	64.35	64.01	2626.05
+Qwen3-8B				
+SFT	61.29	62.99	63.33	2204.57
+SFT+RL	61.97	**65.20**	**65.70**	2348.49

Table 2: Performance comparison of different models in generating PersonaTree memory operations. DS-V3.1, Longcat and DS-R1-0528 respectively denote DeepSeek-V3.1, Longcat-Flash-Chat and DeepSeek-R1-0528.

4.3 Ablation Study

Effectiveness of Components. As presented in Table 1, under the Qwen3-8B-RL setting we conduct a component-level ablation of the adaptive generation pipeline by comparing three inference routes: using only the lightweight fast mode (w/ PersonaTree), adding router-based triggering on top of the fast mode (w/ PersonaTree+Router), and the full agentic recall and fusion (PersonaTree-ALL). The results consistently indicate that PersonaTree alone yields robust gains, but agentic recall and fusion is critical for achieving the best performance, while the routing mechanism can

closely match the full-mode performance while substantially reducing additional retrieval overhead. These gains suggest that directly injecting structured memory already covers a large portion of stable attribute-related requirements, whereas routing and agentic recall further strengthen fine-grained characterization of complex intents and evidence completion.

Enhancement through Training. As shown in Table 2, we further validate the effectiveness of training the memory-operation model. We construct two datasets, PersonaMem 15K and HaluMem 13K. For RL, we use HaluMem 13K for SFT warm-up, and additionally employ 0.5K PersonaMem for process-reward training. For the pure SFT setting, we train on PersonaMem 15K. The results show that training can substantially improve the usability of the memory encoded in PersonaTree as well as downstream reasoning quality. More importantly, whereas ALL Dialogue requires approximately 32K context, the trained PersonaTree introduces only about 2.2K–2.6K tokens of memory context on average, yet can match or exceed the ALL Dialogue baseline on Longcat-Flash-Chat and DeepSeek-R1-0528.

Extraction Models	Direct	Evaluation Models		
		DS-V3.1	Longcat	DS-R1-0528
DeepSeek-V3.1	No	58.91	60.78	58.74
	Yes	**60.03**+1.12	**62.31**+1.53	**61.80**+3.06
Longcat-Flash-Chat	No	58.57	59.59	58.40
	Yes	**58.91**+0.34	**61.63**+2.04	**60.78**+2.38
DeepSeek-R1-0528	No	59.76	60.95	62.31
	Yes	**60.61**+0.85	**63.33**+2.38	**63.50**+1.19

Table 3: Performance comparison between direct Generation and extract-then-transform approaches for generating PersonaTree memory operations.

Selection of Generation Strategies. We compare two strategies for generating PersonaTree operations: *direct generation* (the model directly outputs a tree-operation sequence conditioned on the dialogue) and *extract-then-transform* (first extracting personalized information from the dialogue and then mapping it into tree operations). As shown in Table 3, direct generation achieves consistent advantages in all combinations, with improvements ranging from approximately $+0.34$ to $+3.06$. The two-stage strategy accumulates errors in the intermediate representation, often losing fine-grained semantic and temporal cues needed for accurate tree operations. Based on this finding, we adopt direct generation as the default PersonaTree operation-generation approach in the remainder of this paper.

4.4 Hyperparameter Analysis

Evaluation Models	Chunk Window Size						
	1	3	5	7	10	13	15
`DeepSeek-R1-0528 (Tree Extraction)`							
DeepSeek-V3.1	59.42	60.61	58.74	58.57	60.10	59.25	58.74
Longcat-Flash-Chat	62.82	63.33	62.48	62.65	62.65	62.31	62.48
DeepSeek-R1-0528	60.95	63.50	59.93	60.27	59.25	62.65	61.29
`Qwen2.5-7B-RL (Tree Extraction)`							
DeepSeek-V3.1	60.78	62.82	58.91	61.12	59.25	57.56	59.59
Longcat-Flash-Chat	62.14	64.35	63.67	61.29	64.18	62.82	61.97
DeepSeek-R1-0528	62.82	64.01	60.78	62.82	60.95	61.29	62.14

Table 4: Sensitivity analysis on the size of segmented dialogue chunks.

We analyze the impact of the dialogue chunking window on the quality of tree-operation generation, where each dialogue chunk consists of w consecutive dialogue turns. As shown in Table 4, regardless of whether DeepSeek-R1-0528 or Qwen2.5-7B-RL is used as the tree-operation generation model, $w = 3$ yields the most stable and overall best performance across all three evaluation models. When the window is too small ($w = 1$), chunking becomes overly fragmented and tends to introduce noisy writes; when the window is too large ($w \geq 10$), the within-chunk information density increases substantially, with more frequent cross-topic mixing and timeline collapsing, making critical cues more likely to be diluted or missed. Accordingly, we adopt $w = 3$ for dialogue chunking and tree-operation generation in our experiments.

5 Conclusion

This paper studies memory evolution for long-term personalized dialogue in memory systems and proposes the **Inside Out** framework, which uses an explicit, structured **PersonaTree** as long-term memory to maintain personalized states under unbounded interactions. Concretely, we build a hierarchical schema grounded in the biopsychosocial model and develop an iterative tree-update mechanism. We then train a lightweight **MemListener** with process-reward RL to compress unstructured dialogue streams into executable tree operations. At inference time, we design an adaptive generation pipeline. Experiments show that PersonaTree-driven personalization consistently outperforms existing baselines across multiple response models, and further highlight the potential of using small models for memory maintenance.

Limitations

This work focuses on the structured evolution of long-term personalized memory, and the current implementation and empirical validation delineate clear directions for future extension:

Scope of the schema and PersonaTree. We adopt a hierarchical schema grounded in the biopsychosocial model to define the writable space and capacity constraints, yielding a consistent and controllable representation of long-term memory. For finer-grained domain knowledge, task-skill profiles, or cross-domain user states, the schema can be further extended into composable subtrees or plugin-style modules to accommodate broader application needs.

Applicability of the memory-evolution strategy. PersonaTree is iteratively updated via atomic tree operations, with add/modify/delete semantics uniformly expressed as trunk and leaf-level text rewrites. This abstraction is effective for preserving stable core traits and compressible summaries; for memory forms requiring stronger temporal constraints, evidence provenance, or multi-version coexistence, additional metadata such as explicit timestamps, confidence scores, and source pointers can be incorporated to improve traceability and controllability.

Engineering extensions for privacy and governance. As an explicit long-term memory carrier, PersonaTree naturally supports access control, interpretable edits, and revocability. For real-world deployment, it can be further complemented with user-facing memory management, sensitivity-aware field stratification, and data-minimization storage policies to meet stricter governance requirements.

References

Aili Chen, Chengyu Du, Jiangjie Chen, Jinghan Xu, Yikai Zhang, Siyu Yuan, Zulong Chen, Liangyue Li, and Yanghua Xiao. 2025a. Deeper insight into your user: Directed persona refinement for dynamic persona modeling. *arXiv preprint arXiv:2502.11078*.

Ding Chen, Simin Niu, Kehang Li, Peng Liu, Xiangping Zheng, Bo Tang, Xinchi Li, Feiyu Xiong, and Zhiyu Li. 2025b. Halumem: Evaluating hallucinations in memory systems of agents. *arXiv preprint arXiv:2511.03506*.

Yi-Pei Chen, Noriki Nishida, Hideki Nakayama, and Yuji Matsumoto. 2024. Recent trends in personalized dialogue generation: A review of datasets, methodologies, and evaluations. *arXiv preprint arXiv:2405.17974*.

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*.

Xu Han, Bin Guo, Yoon Jung, Benjamin Yao, Yu Zhang, Xiaohu Liu, and Chenlei Guo. 2023. Personapkt: Building personalized dialogue agents via parameter-efficient knowledge transfer. *arXiv preprint arXiv:2306.08126*.

Qiushi Huang, Shuai Fu, Xubo Liu, Wenwu Wang, Tom Ko, Yu Zhang, and Lilian Tang. 2023. Learning retrieval augmentation for personalized dialogue generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2523–2540.

Bowen Jiang, Zhuoqun Hao, Young-Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo J Taylor, and Dan Roth. 2025. Know me, respond to me: Benchmarking llms for dynamic user profiling and personalized responses at scale. *arXiv preprint arXiv:2504.14225*.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. 2025. Memory os of ai agent. *arXiv preprint arXiv:2506.06326*.

Guanrong Li, Xinyu Liu, Zhen Wu, and Xinyu Dai. 2025a. Persona-aware alignment framework for personalized dialogue generation. *Transactions of the Association for Computational Linguistics*, 13:1722–1742.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and William B Dolan. 2016. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003.

Zhiyu Li, Shichao Song, Chenyang Xi, Hanyu Wang, Chen Tang, Simin Niu, Ding Chen, Jiawei Yang, Chunyu Li, Qingchen Yu, Jihao Zhao, Yezhaohui Wang, Peng Liu, Zehao Lin, Pengyuan Wang, Jiahao Huo, Tianyi Chen, Kai Chen, Kehang Li, and 20 others. 2025b. Memos: A memory os for ai system. *arXiv preprint arXiv:2507.03724*.

Lei Liu, Xiaoyan Yang, Yue Shen, Binbin Hu, Zhiqiang Zhang, Jinjie Gu, and Guannan Zhang. 2023. Think-in-memory: Recalling and post-thinking enable llms with long-term memory. *Preprint*, arXiv:2311.08719.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Andrea Madotto, Zhaojiang Lin, Chien-Sheng Wu, and Pascale Fung. 2019. Personalizing dialogue agents via meta-learning. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 5454–5459.

Hazel Rose Markus. 1977. Self-schemata and processing information about the self. *Journal of Personality and Social Psychology*, 35:63–78.

Kai Tzu-iunn Ong, Namyoung Kim, Minju Gwak, Hyungjoo Chae, Taeyoon Kwon, Yohan Jo, Seungwon Hwang, Dongha Lee, and Jinyoung Yeo. 2025. Towards lifelong dialogue agents via timeline-based memory management. 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 8631–8661.

Atsushi Otsuka, Kazuya Matsuo, Ryo Ishii, Narichika Nomoto, and Hiroaki Sugiyama. 2024. User-specific dialogue generation with user profile-aware pre-training model and parameter-efficient fine-tuning. *arXiv preprint arXiv:2409.00887*.

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

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, Bangkok, Thailand. Association for Computational Linguistics.

Zhaoxuan Tan and Meng Jiang. 2023. User modeling in the era of large language models: Current research and future directions. *Preprint*, arXiv:2312.11518.

Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. 2024. Democratizing large language models via personalized parameter-efficient fine-tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6476–6491, Miami, Florida, USA. Association for Computational Linguistics.

Piiastiina Tikka and Harri Oinas-Kukkonen. 2019. Tailoring persuasive technology: A systematic review of literature of self-schema theory and transformative learning theory in persuasive technology context.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*.

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

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making retrieval-augmented language models robust to irrelevant context. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. 2024. Swift:a scalable lightweight infrastructure for fine-tuning. *Preprint*, arXiv:2408.05517.

Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. 2019. Personalized dialogue generation with diversified traits. *arXiv preprint arXiv:1901.09672*.

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 Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

A Rethinking Personalization

In Human-AI interaction, building agents capable of deeply personalized dialogue has long been a central goal. However, the dominant research paradigm largely concentrates on personalization via explicit profiles (Li et al., 2016; Zhang et al., 2018). Under this setting, researchers typically provide either a structured or unstructured persona description, or a set of persona-related texts to be retrieved, and the model is tasked with generating user-aligned responses conditioned on this static, pre-specified information (Huang et al., 2023; Li et al., 2025a).

Although this paradigm offers advantages in controllability and evaluation convenience, it deviates substantially from real-world interpersonal interaction. This deviation is mainly reflected in:

- **Misalignment of information sources:** In everyday life, our understanding of a person's traits rarely comes from a self-introduction document; instead, it is implicitly and dynamically constructed from long-term interaction history (Han et al., 2023; Otsuka et al., 2024).
- **Limited characterization of persona:** Explicit persona descriptions are often highly abstracted and simplified, failing to capture subtle linguistic styles, background knowledge,

distinctive interaction patterns, and affective dynamics that emerge in authentic conversations (Chen et al., 2025a).

- **A static assumption of persona:** Real personality traits and linguistic styles vary across contexts and conversations, whereas static-profile approaches struggle to model such adaptive dynamics (Ong et al., 2025).

Given these limitations, our study targets a more challenging and more realistic core problem: *How can a model, relying solely on a long personalized dialogue history, learn and emulate one participant's implicit persona to generate responses that remain consistent in style, content, and relational stance?*

Our motivation is to help bridge this gap, with three primary implications:

- **Improving the realism of personalized dialogue:** enabling a shift from role-playing to faithful imitation, producing responses that are more natural, credible, and person-like.
- **Advancing deep personalization modeling:** moving beyond understanding facts about a person toward modeling how a person becomes who they are.
- **Expanding real-world applicability:** in emerging applications such as personalized assistants and affective companions, the ability to reproduce individual styles from historical data is crucial.

B MemListener Training

After the SFT warm-up, we continue alignment with the remaining data using RL driven by process-based rewards. We set the model's maximum context length to 11K tokens, with the input length capped at 10K, which constitutes a typical ultra-long sequence optimization setting. If we were to adopt sample-level group-relative policy optimization (GRPO), the key decision signals in long sequences would be easily diluted by within-sample averaging; moreover, when group-wise sampling under the same input yields outputs that are all correct or all incorrect, the advantage term degenerates, resulting in insufficient effective gradients. Consequently, training stability and sample efficiency are constrained.

Therefore, the RL stage is conducted using the Swift RLHF framework (Zhao et al., 2024) with the DAPO algorithm (Table 5). All model parameters are updated via full fine-tuning, and training is performed in bfloat16 precision to balance numerical stability and memory efficiency. We further employ a dynamic, LLM-as-a-judge evaluation strategy, using Qwen3-32B (reasoning mode) as the discriminator to score the gap between the model's prediction and the ground truth. The judge is prompted using the template in Table 10, and its assessment signal is used to guide optimization during RL. For transparency and reproducibility, we release the complete training scripts in our public repository.

Parameter	**Value**
Training type	Full fine-tuning
Model precision	bfloat16
Learning rate	1×10^{-6}
Per Device Train Batch Size	1
Training batch size	1
Gradient accumulation steps	8
Number of epochs	1
Warmup ratio	0.01
Max gradient norm	1.0
Max sequence length	11264
Max generation length	512
Number of generations	8
Temperature	1.0
Top-p	0.9
Top-k	50
Clipping ϵ	0.2 / 0.28
β (KL control)	0.001
Dynamic sampling	Enabled
Max resample times	3

Table 5: Key hyperparameters for DAPO training.

A learning rate of 1×10^{-6} is adopted, together with a warmup ratio of 0.01. Due to memory constraints, the per-device training batch size is set to 1, while the effective batch size is increased using gradient accumulation over 8 steps. Gradient norms are clipped to 1.0 to ensure stable optimization. Training is performed for a single epoch.

For each input prompt, 8 candidate responses are sampled with a maximum generation length of 512 tokens and a maximum context length of 11,264 tokens. Stochastic decoding is controlled using temperature = 1.0, top-p = 0.9, and top-k = 50. Policy updates use asymmetric clipping with $\epsilon = 0.2$ and $\epsilon_{\text{high}} = 0.28$.

A KL-control coefficient β is introduced to regulate the divergence between the optimized policy and the reference model. Larger β values enforce stronger regularization toward the reference policy. In our GRPO-based setting, β is set to 0.001. Dynamic sampling is enabled to enhance response diversity, with the maximum number of resampling attempts limited to 3.

During training data construction, we select subsets from HaluMem and PersonaMem that are relevant to implicitly characterizing user-specific attributes as the raw corpus sources. Using the dynamic PersonaTree evolution procedure described in Section 3.2 as the backbone, we invoke DeepSeek-R1-0528 to generate supervision signals for training. Concretely, for each dialogue segment, we prompt the generator to produce an executable operation sequence and its corresponding post-update tree state under the given schema and update constraints, thereby mapping raw dialogues into structured samples with ground-truth operations and versioned tree evolution. To control noise and spurious correlations, we manually verify the synthesized samples and filter out those with invalid operation syntax, incorrect path references, or semantically inconsistent writes. The specific prompts are shown in Table 10.

C Extra Experiments

C.1 Baselines

To evaluate the effectiveness of our proposed approach, we compared it against six baseline methods. These include two standard interaction paradigms (Only LLM and Full Context) and four state-of-the-art memory management frameworks designed for LLM-based agents. All memory management frameworks were evaluated under their officially recommended best configurations.

Only LLM. As a foundational baseline, we employ the LLM directly without providing any historical conversation data. In this setting, the model operates in a stateless manner, relying solely on its pre-trained parametric knowledge and internal reasoning capabilities to address user queries. This method serves as a lower bound, isolating the model's intrinsic commonsense reasoning from its ability to recall specific interactional details.

Full Context. This method involves concatenating the entire chronological history of the conversation and inputting it into the LLM's context window for every interaction. By providing the model with complete access to all prior dialogue, this approach serves as a theoretical upper bound for retrieval accuracy within the limits of the model's context window. However, it effectively ignores the challenges of memory selection and computational efficiency.

LangMem. LangMem is a framework designed to enable agents to learn and adapt through continuous interactions. It provides a suite of functional primitives that allow agents to manage memory within the active conversational flow ("hot path") while utilizing a background manager to asynchronously extract, consolidate, and update knowledge. LangMem integrates natively with the LangGraph ecosystem, offering a core memory API that supports prompt refinement and long-term consistency across sessions. By separating immediate memory management tools from background consolidation processes, it aims to maintain consistent agent behavior without increasing latency during inference.

Mem0 (Chhikara et al., 2025). Mem0 addresses the limitations of fixed context windows by introducing a scalable, memory-centric architecture. It employs a multi-level memory structure that retains User, Session, and Agent states to facilitate adaptive personalization. A key feature of Mem0 is its utilization of graph-based memory representations to capture complex relational structures between conversational elements. This approach allows for the dynamic extraction and retrieval of salient information, optimizing for both latency and token cost. Mem0 is designed to be production-ready, focusing on reducing the computational overhead typically associated with full-context processing while maintaining high retrieval accuracy in multi-session dialogues.

A-Mem (Agentic Memory) (Xu et al., 2025). A-Mem proposes a self-organizing memory system inspired by the Zettelkasten knowledge management method. Unlike traditional static storage, A-Mem enables agents to dynamically organize memories through intelligent indexing and linking. When new information is ingested, the system generates comprehensive notes with structured attributes—such as contextual descriptions and tags—and establishes connections with historical data. A distinctive feature of A-Mem is its support for "memory evolution," where the integration of new experiences can trigger updates to the repre-

Methods	Overall	Recall-Facts	Pref-Rec	New-Ideas	Recall-Reason	Pref-Evol	Gen-New	Recall-User
DeepSeek-V3.1 (Response)								
Only LLM	52.63	59.69	43.64	5.38	80.81	65.47	42.11	52.94
ALL Dialogue	63.84	78.29	61.82	12.90	81.82	65.47	77.19	**76.47**
LangMem+DeepSeek-V3.1	56.37	72.87	52.73	9.68	78.79	61.15	50.88	47.06
Mem0+DeepSeek-V3.1	61.80	82.95	47.27	7.53	83.84	63.31	71.93	70.59
A-Mem+DeepSeek-V3.1	60.61	76.74	43.64	6.45	85.86	64.03	71.93	**76.47**
MemoryOS+DeepSeek-V3.1	61.97	75.19	**67.27**	9.68	80.81	63.31	73.68	70.59
PersonaTree-ALL+DeepSeek-V3.1	**70.80**	**88.37**	63.64	**24.73**	**86.87**	**68.35**	**91.23**	70.59
Longcat-Flash-Chat (Response)								
Only LLM	54.33	62.02	43.64	8.60	84.85	**71.22**	28.07	52.94
ALL Dialogue	61.80	79.07	63.64	15.05	87.88	53.96	68.42	70.59
LangMem+DeepSeek-V3.1	57.39	75.97	50.91	10.75	79.80	63.31	45.61	52.94
Mem0+DeepSeek-V3.1	60.27	77.52	54.55	9.68	85.86	64.03	50.88	**76.47**
A-Mem+DeepSeek-V3.1	60.44	80.62	50.91	8.60	85.86	66.19	47.37	70.59
MemoryOS+DeepSeek-V3.1	64.52	79.84	65.45	16.13	87.88	64.75	66.67	64.71
PersonaTree-ALL+DeepSeek-V3.1	**73.34**	**88.37**	**69.09**	**31.18**	**91.92**	69.78	**87.72**	**76.47**
DeepSeek-R1-0528 (Response)								
Only LLM	44.14	30.23	38.18	19.35	76.77	55.40	38.60	41.18
ALL Dialogue	64.86	69.77	63.64	11.83	84.85	73.38	84.21	70.59
LangMem+DeepSeek-V3.1	56.54	62.79	54.55	18.28	78.79	66.19	49.12	41.18
Mem0+DeepSeek-V3.1	47.71	32.56	52.73	21.51	73.74	57.55	47.37	58.82
A-Mem+DeepSeek-V3.1	47.03	27.13	45.45	**27.96**	76.77	55.40	43.86	76.47
MemoryOS+DeepSeek-V3.1	61.97	64.34	**76.36**	11.83	82.83	66.19	73.68	76.47
PersonaTree-ALL+DeepSeek-V3.1	**74.53**	**81.40**	70.91	26.88	**92.93**	**82.01**	**87.72**	**82.35**

Table 6: Extra experimental results are presented on three different response models. Recall-Facts, Pref-Rec, New-Ideas, Recall-Reason, Pref-Evol, Gen-New, and Recall-User respectively denote recalling user-shared facts, providing preference-aligned recommendations, suggesting new ideas, recalling reasons behind preference updates, tracking preference evolution, generalizing to new scenarios, and recalling user-mentioned facts. **Bold** and underlined numbers denote the best and second-best results, respectively.

sentations of existing memories. This agent-driven mechanism allows the memory network to continuously refine its structure and understanding over time.

MemoryOS (Kang et al., 2025). Drawing inspiration from operating system principles, MemoryOS introduces a hierarchical storage architecture designed to manage agent memory comprehensively. The system comprises four core modules: Storage, Updating, Retrieval, and Generation. It organizes memory into three distinct levels: short-term, mid-term, and long-term personal memory. To manage data flow between these levels, MemoryOS employs specific strategies such as a dialogue-chain-based First-In-First-Out (FIFO) principle for short-to-mid-term updates and a segmented page organization strategy for mid-to-long-term consolidation. This hierarchical approach aims to maximize context coherence and personalization by mimicking the efficient resource management found in traditional operating systems.

C.2 Extra Experiments Results

To validate the robustness of our approach for the tree-operation generator, we further adopt DeepSeek-V3.1 as a unified extraction model to conduct comparative evaluations across all methods, and we report results separately under three different response models (DeepSeek-V3.1, Longcat-Flash-Chat, and DeepSeek-R1-0528; see Table 6) to minimize potential biases and hallucination effects introduced by any particular generation component. The results show that, even when replacing the extraction model DeepSeek-R1-0528 with DeepSeek-V3.1, our method (PersonaTree-ALL) still achieves the best Overall scores under all three response models, reaching 70.80, 73.34, and 74.53, respectively. This corresponds to improvements of +18.17/+19.01/+30.39 over Only LLM and +6.96/+11.54/+9.67 over ALL Dialogue, and it substantially outperforms the representative memory baseline MemoryOS by +8.83/+8.82/+12.56. From a metric-wise perspective, the gains are particularly pronounced on dimensions requiring stronger detail completion and open-ended generation; for example, New-Ideas improves over ALL Dialogue by +11.83 (24.73 vs. 12.90), +16.13 (31.18 vs. 15.05), and +15.05 (26.88 vs. 11.83), respectively. Meanwhile, Recall-Facts, Pref-Rec, and Gen-New also exhibit consistent improvements. These findings indicate that PersonaTree's structured memory representation and retrieval-augmented generation

mechanism do not rely on any specific extraction model; rather, the benefits transfer stably across extractors and response models, further supporting the generality and effectiveness of the proposed method.

Ablation Study. Beyond the partial ablation studies reported in Table 1, we further conduct a more comprehensive ablation analysis on the remaining configurations in Tables 1 and 6, with detailed results presented in Tables 7 and 8. Overall, these additional ablation findings are consistent with the observations in Table 1, further corroborating the effectiveness and contributions of the key components across different experimental settings, and providing stronger empirical support for the main conclusions in Section 4.3.

C.3 Visualization

To more intuitively illustrate the differences among the methods in Tables 1 and 6 across diverse capability dimensions, we further provide radar-chart visualizations (see Figure 2), corresponding to two memory-extraction settings: (a) using DeepSeek-R1-0528 as the extractor; and (b) using DeepSeek-V3.1 as the extractor. For each setting, we conduct a unified comparison under three response models: DeepSeek-V3.1, Longcat-Flash-Chat, and DeepSeek-R1-0528. Overall, PersonaTree exhibits a more outward-expanded polygonal profile across both extraction settings and all three response models, indicating that its gains are not concentrated on a single metric but instead span multiple dimensions, including factual recall, preference consistency, preference evolution, and new-content generation.

Moreover, the trends across the two subplots are highly consistent, suggesting that PersonaTree's improvements are robust to the choice of memory-extraction model: even when the extractor is replaced, the method maintains stable advantages across the multi-dimensional metrics.

On the other hand, Figure 3 reports the overall performance of different memory-operation models under three response models. We observe that untrained extractors generally lag behind trained counterparts, while the two-stage training paradigm (SFT+RL) yields stable and transferable improvements. Meanwhile, compared with ALL Dialogue, which requires substantially longer context, the trained PersonaTree achieves higher accuracy with only a relatively short memory context, further highlighting the advantages of structured memory in terms of information compression and utilization efficiency.

Figure 4 characterizes how the dialogue chunking window size w influences performance. Across settings, the curves consistently exhibit the pattern that a moderate window is optimal. In particular, values around $w = 3$ are more stable and overall superior under all three response models, suggesting that this configuration strikes a more appropriate trade-off between contextual sufficiency and update frequency.

D Initial PersonaTree Instance

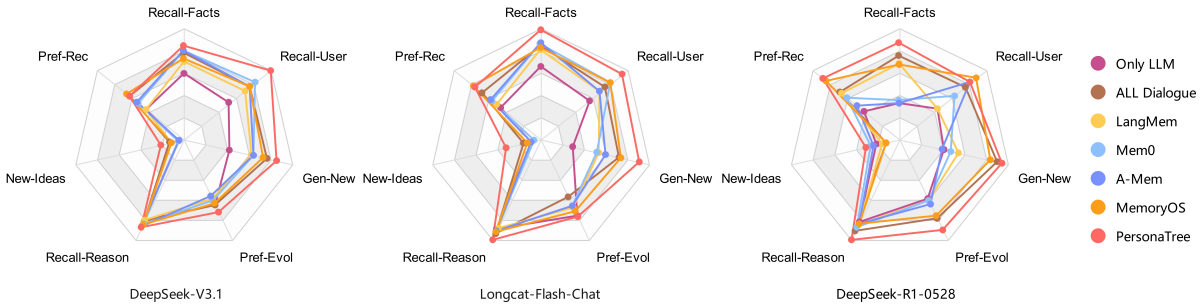
To delineate the theoretical boundaries of the memory tree, we construct a hierarchical Schema based on the Biopsychosocial model, scientifically decomposing user characteristics into three core dimensions: (1) Biological Aspects: Establishes biological traits by referencing theories in human morphology, human physiology, chronobiology, etc. (2) Psychological Aspects: Deeply mines psychological and characterological traits through cognitive psychology, the Big Five personality theory, metacognition theory, etc. (3) Social Aspects: Unifies demographic and behavioral traits based on social identity theory, behavioral psychology, attachment theory, etc. Due to the lengthy initial personalization tree, we include the full schema in our codebase, where the complete content can be inspected.

Methods	Overall	Recall-Facts	Pref-Rec	New-Ideas	Recall-Reason	Pref-Evol	Gen-New	Recall-User
DeepSeek-R1-0528 (Tree Extraction)								
+DeepSeek-V3.1 (Response)								
PersonaTree-ALL	**71.14**	**88.37**	56.36	**22.58**	**88.89**	**71.22**	**89.47**	**88.24**
PersonaTree+Router	69.61	87.60	**60.00**	17.20	**88.89**	69.06	85.96	**88.24**
Only PersonaTree	60.61	77.52	49.09	6.45	78.79	66.19	73.68	70.59
+Longcat-Flash-Chat (Response)								
PersonaTree-ALL	**72.67**	88.37	**72.73**	**26.88**	**92.93**	69.78	**85.96**	64.71
PersonaTree+Router	70.46	**89.15**	60.00	19.35	87.88	**71.94**	80.70	**94.12**
Only PersonaTree	63.33	75.97	60.00	11.83	88.89	66.91	70.18	58.82
+DeepSeek-R1-0528 (Response)								
PersonaTree-ALL	**74.87**	80.62	69.09	**27.96**	**94.95**	82.01	**89.47**	**82.35**
PersonaTree+Router	73.68	**82.17**	**74.55**	23.66	92.93	**83.45**	84.21	52.94
Only PersonaTree	63.50	67.44	63.64	16.13	88.89	69.06	75.44	58.82
Qwen2.5-7B-rl (Tree Extraction)								
+DeepSeek-V3.1 (Response)								
PersonaTree-ALL	**71.31**	**89.15**	**61.82**	23.66	**91.92**	69.06	**85.96**	**76.47**
PersonaTree+Router	70.80	87.60	60.00	**25.81**	87.88	**71.22**	**85.96**	70.59
Only PersonaTree	62.82	77.52	**61.82**	8.60	89.90	63.31	70.18	64.71
+Longcat-Flash-Chat (Response)								
PersonaTree-ALL	**73.34**	**93.02**	**67.27**	**27.96**	**92.93**	67.63	**89.47**	70.59
PersonaTree+Router	72.33	92.25	**67.27**	23.66	91.92	**68.35**	85.96	**76.47**
Only PersonaTree	64.35	79.07	63.64	11.83	88.89	65.47	70.18	70.59
+DeepSeek-R1-0528 (Response)								
PersonaTree-ALL	**74.70**	79.84	**80.00**	27.96	**94.95**	**82.01**	**84.21**	64.71
PersonaTree+Router	74.19	**80.62**	76.36	**30.11**	93.94	80.58	**84.21**	58.82
Only PersonaTree	64.01	68.99	63.64	12.90	84.85	71.22	80.70	**70.59**

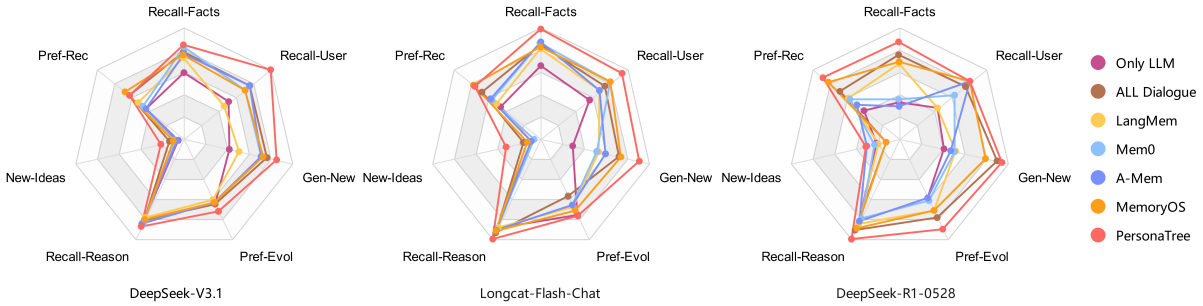
Table 7: Ablation study of PersonaTree components. Recall-Facts, Pref-Rec, New-Ideas, Recall-Reason, Pref-Evol, Gen-New, and Recall-User respectively denote recalling user-shared facts, providing preference-aligned recommendations, suggesting new ideas, recalling reasons behind preference updates, tracking preference evolution, generalizing to new scenarios, and recalling user-mentioned facts. **Bold** numbers indicate the best results within each group.

Methods	Overall	Recall-Facts	Pref-Rec	New-Ideas	Recall-Reason	Pref-Evol	Gen-New	Recall-User
DeepSeek-V3.1 (Tree Extraction)								
+DeepSeek-V3.1 (Response)								
PersonaTree-ALL	**70.80**	**88.37**	**63.64**	**24.73**	86.87	68.35	**91.23**	**70.59**
PersonaTree+Router	69.61	85.27	61.82	18.28	**89.90**	**70.50**	87.72	**70.59**
Only PersonaTree	60.03	74.02	56.60	6.59	82.47	62.50	71.93	64.71
+Longcat-Flash-Chat (Response)								
PersonaTree-ALL	**73.34**	88.37	**69.09**	**31.18**	**91.92**	**69.78**	**87.72**	**76.47**
PersonaTree+Router	70.63	**91.47**	63.64	21.51	89.90	67.63	84.21	70.59
Only PersonaTree	62.31	78.29	54.55	10.75	86.87	66.91	63.16	64.71
+DeepSeek-R1-0528 (Response)								
PersonaTree-ALL	**74.53**	**81.40**	70.91	**26.88**	**92.93**	**82.01**	**87.72**	**82.35**
PersonaTree+Router	73.51	**81.40**	**76.36**	24.73	**92.93**	79.14	84.21	76.47
Only PersonaTree	61.80	68.22	52.73	13.98	82.83	71.94	73.68	58.82

Table 8: Ablation Study II of PersonaTree Components. Recall-Facts, Pref-Rec, New-Ideas, Recall-Reason, Pref-Evol, Gen-New, and Recall-User respectively denote recalling user-shared facts, providing preference-aligned recommendations, suggesting new ideas, recalling reasons behind preference updates, tracking preference evolution, generalizing to new scenarios, and recalling user-mentioned facts. **Bold** numbers indicate the best results within each group.



(a) DeepSeek-R1-0528 for memory extraction, with PersonaTree powered by Qwen3-8B-RL.



(b) DeepSeek-V3.1 for memory extraction, with PersonaTree powered by Qwen3-8B-RL.

Figure 2: Radar-chart comparison of PersonaTree and baselines across multi-dimensional capability metrics under two memory-extraction settings (DeepSeek-R1-0528, DeepSeek-V3.1) and three response models (DeepSeek-V3.1, Longcat-Flash-Chat, DeepSeek-R1-0528)

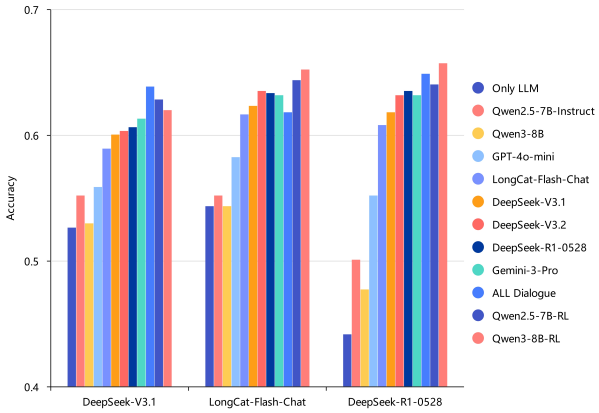
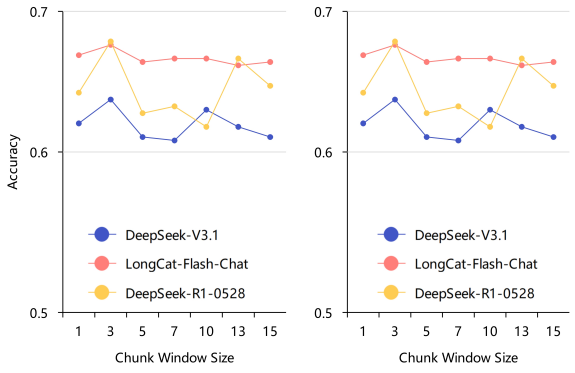


Figure 3: Overall performance of memory-operation models across three response models.



(a) Deepseek-R1-0528 for Memory Extraction (b) Qwen2.5-7B-RL for Memory Extraction

Figure 4: Effect of dialogue chunking window size on performance across three response models.

Prompts for Operational Generation in PersonaTree

You are a Memory-Tree Operation Generator. You will be given:
(1) An initial persona schema represented as a hierarchical JSON tree.
(2) A dialogue history.\n\n
Your objective is to transform the dialogue history into a sequence of operations for updating the persona schema, **covering as comprehensively as possible all information about this person, especially personalized characteristics**.\n\n
About the schema:
- The schema below contains **user attribute information that has already been successfully structured**;
- Treat the schema as "recorded information" and **do not re-extract fields that already exist**;
- Generate operations for the schema only when the dialogue history introduces additional facts, details, or preferences not yet covered by the schema;
- If the dialogue history conflicts with the schema, the **most recent explicit statement** in the dialogue should prevail.\n\n
Principles for using ADD / UPDATE / DELETE / NO_OP:
* Use: ADD(path, "value") when an attribute at that path has **not been recorded at all**. Prefer creating more branches and avoid overly long content in a single attribute.
* Use: UPDATE(path, "value") when an attribute at that path already has a record and the current passage **supplements, refines, or corrects** it.
* Use: DELETE(path, None) only when the passage explicitly states that an existing piece of information **is no longer valid, is negated, or should be removed**.
* If the passage does not entail any changes, output a single line: NO_OP().\n\n
Key requirements for "value" in UPDATE (very important):
* "value" must semantically **contain or integrate the previously valid information** while incorporating or reflecting the new information, yielding a more complete, more accurate, and up-to-date description.
* It is **strictly forbidden** to discard useful original content and keep only the new information in an UPDATE.
* When the new information is supplemental or more specific, the value should be an integrated expression of "original information + new supplementation".
* When the new information conflicts with the old, the value should describe the "current latest and most reasonable state", while retaining non-conflicting old details whenever possible.\n\n
Notes:\n1. Treat each leaf node in the JSON schema as an attribute slot capable of storing a textual value.\n\n
2. For each distinct user personal attribute mentioned in the dialogue history:
* Locate the most closely matching and most specific leaf node in the schema.
* Generate **exactly one and only one** operation for that attribute.\n\n
3. You may use only the following operations:
* ADD(path, "value"), UPDATE(path, "value"), DELETE(path, None), NO_OP()\n\n
4. Requirements for the "path" format:
* Use a JSON key path separated by English periods. Example:
1_Biological_Characteristics.Physiological_Status.Age_Related_Characteristics.Chronological_Age\n\n
5. Requirements for the "value" format:
* Provide a natural-language expression extracted from or normalized based on the dialogue history.
* It must be enclosed in English double quotation marks.\n\n
6. Output format (must be strictly followed):
* Output only operations, one operation per line. * Do not add any explanations or comments. * The only permissible forms are: ADD(<path>, "<value>"), UPDATE(<path>, "<value>"), DELETE(<path>, None), NO_OP()\n\n
Persona Schema:\n\n{schema}\n\n
Dialogue History:\n\n{dialogue_text}\n\n
Now, based on the given dialogue history, output only the operations:

Table 9: Prompt for operational generation in PersonaTree for training and inference.

Reward-Function Prompt

You are a strict "overall scorer for attribute-tree operations". Your task is to assign an overall quality score in [-1, 1] to the model-predicted operation sequence Pred_Ops, given the ground-truth annotated operation sequence GT_Ops.

[Input]
- GT_Ops (ground truth): a list of operations, where each element is of the form ADD(path, value) / UPDATE(path, value) / DELETE(path, value) / NO_OP()
- Pred_Ops (prediction): a list of operations in the same format as above

[Critical Constraints]
1) Output only a single JSON object: {"score": <float>}. Do not output any explanation and do not include any extra fields.
2) score must be a continuous floating-point number within [-1, 1] (any value is allowed). It is recommended to keep 2 decimal places.
3) The "score-tier reference" below serves only as anchors for aligning overall quality. You should fine-tune between anchors to output a more granular score.
4) For example, if the overall quality falls between 0.7 and 1.0, output a value in [0.71, 0.99]; if it falls between 0.5 and 0.7, output a value in [0.51, 0.69]; and so on.

[Score-Tier Reference (Overall Quality Anchors)]
* 1.0 (nearly perfect): Pred and GT are almost entirely consistent on key operations; types/paths are nearly identical; values are semantically equivalent; no redundant operations.
* 0.7 (high quality): most key operations are correct; only minor value-level deviations, or very few missing/redundant operations.
* 0.5 (moderately usable): the overall approach and core direction are correct; some missing/redundant operations exist; some paths/values are incorrect, but the main semantics are not affected.
* 0.3 (partially reliable): about half of the content is reliable; some key operations are correct while others are wrong, requiring some fixes.
* 0.0 (slightly correct): only a small number of operations or fragments are correct; missing/redundant operations and errors are evident; key operations are mixed correct/incorrect.
* -0.3 (barely relevant): broadly related but with many omissions/errors; it is only apparent that the model is attempting the task, and it is essentially unusable as-is.
* -0.5 (clearly off-target): most key operations are missing or incorrect; many wrong paths/types or obviously redundant operations; overall deviates from expectations.
* -0.7 (catastrophic): large-scale structural/semantic disorder; almost unusable.
* -1.0 (meaningless output): clearly meaningless, garbage text, or unrelated to the task.

[Output Format]
Output only the JSON object containing the score, with no additional notes or explanations.
Output only:
{"score": <float>}

[Task Data]
- GT_Ops:
{gt_ops}

- Pred_Ops:
{pred_ops}

Table 10: Reward-function prompt for process-reward RL training.