Preferences Shape Responses: Benchmarking Preference Understanding of Large Language Models

Anonymous ACL submission

Abstract

User-LLM interaction histories contain rich user preferences that can guide models toward more personalized responses. However, there still lacks a systematic benchmark for evaluating whether LLMs can automatically infer such preferences from multi-turn interactions and align their responses accordingly. To address this gap, we introduce InterPref, a benchmark for user preference alignment in human-LLM interactions. InterPref contains paired interaction histories with distinct preferences, constructed from real user conversations collected through a self-built temporary website, together with a systematic evaluation toolkit supporting multiple evaluation protocols. We evaluate more than 23 open-source and proprietary LLMs under diverse settings, including vanilla inference, hand-crafted prompts, human-designed workflows, and fine-tuning. Our results show that preference alignment from interaction history remains an overlooked and challenging capability for current LLMs. Through model comparison and failure analysis, we identify key bottlenecks and show that fine-tuning on InterPref improves models' ability to incorporate user preferences. Our study provides a foundation for developing more personalized LLM-based assistants. The project is available at https://anonymous.4open.science/r/InterPref

1 Introduction

Large Language Models (LLMs) have enabled the development of powerful personal assistants (Qiu et al., 2019; He et al., 2020). Users of such assistants exhibit diverse preferences across daily life and professional domains, motivating the need for LLMs to align their responses with individual user preferences. Existing works address this by integrating external memory mechanisms (OpenAI, 2024) or additional context retrieval (Yuan et al., 2024; Ning et al., 2024) to maintain and utilize user-specific information. While these approaches are effective, a fundamental question remains: to what extent can the model's inherent capabilities infer and adapt to user preferences in their context? Specifically, if we provide the interaction history directly as model context, can the LLM automatically generate responses aligned with distinct user preferences?

To study this systematically, we formalize the problem as **Direct Inference-time Preference Alignment (DIPA)**. The goal is to generate personalized responses to a query conditioned solely on prior interaction history. Figure 1 illustrates the core idea: given varying interaction histories, the same query should elicit distinct responses that reflect different user preferences.

To facilitate evaluation, we introduce InterPref, a high-quality dataset derived from real user-LLM conversations. To ensure privacy and evaluation consistency, we generate a synthetic version that preserves the core user preferences while removing sensitive details. This pipeline extracts and consolidates the latent preferences from often sparse and noisy conversations, resulting in a more focused and reliable benchmark.

Assessing preference alignment is inherently challenging due to its subjective nature. We adopt a multi-dimensional evaluation strategy combining absolute and relative metrics. Absolute metrics are obtained via LLM-as-a-judge (Kwok et al., 2024), including **task completion score** for overall response quality and **preference alignment score** for user alignment. To reduce biases in absolute scoring, we introduce **Pass Rate**, evaluating a model's ability to adapt to different preferences, and **Win Rate**, comparing models in identical scenarios (Chiang et al., 2024; Munos et al., 2024).

Evaluating 23 models on InterPref reveals that preference awareness is not inherent to LLMs: without explicit guidance, even strong models achieve only $\sim 20\%$ pass rate. Performance im-

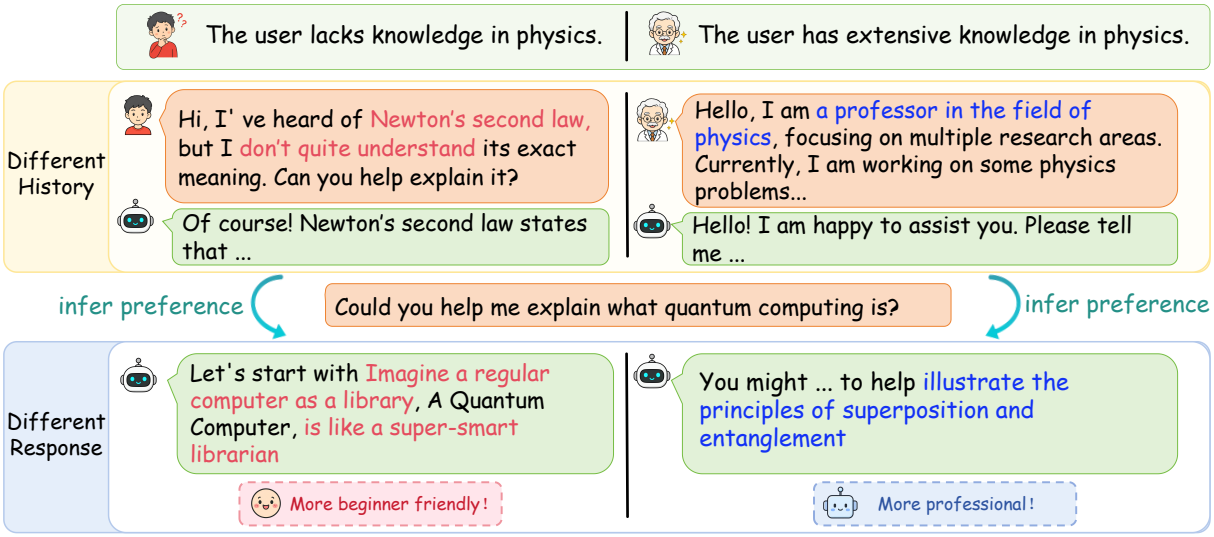

Figure 1: The model understands different preferences embedded within the interaction history and generates responses to the same query that align with distinct user preferences.

proves with model size, but substantial gaps exist across model families (e.g., Llama-3.1 outperforms Qwen-2.5), indicating that preference alignment is largely orthogonal to standard reasoning capabilities (Qwen et al., 2025).

Beyond benchmarking, InterPref also serves as a high-quality resource for capability distillation. By leveraging diverse preferences and queries, we use advanced LLMs to generate refined, aligned responses. Filtering for quality, we construct a distillation dataset to fine-tune smaller models. Experiments show that Llama-3.1-8B, fine-tuned on this dataset, improves its pass rate to 33.9% on real conversations, nearly tripling the baseline of 12.5%, demonstrating both the evaluation and improvement potential of InterPref.

2 Related Works

Inference-time Alignment Benchmarks Previous benchmarks for preference alignment can be broadly divided into two categories. 1) isolated single-turn tasks, such as personalized summarization (Patel et al., 2024), recommendation (Sun and Zhang, 2018), and style imitation (Cho et al., 2025). LaMP (Salemi et al., 2024) integrates many such tasks into a unified benchmark. 2) preference alignment in general multi-turn interactions, where the model must infer user preferences from dialogue history. Our benchmark belongs to this category. The closest work to ours in this direction is PrefEval (Zhao et al., 2025). However, PrefEval relies mainly on LLM-based absolute scoring, which we find to be unstable and insufficiently discriminative: different evaluators often follow inconsistent scoring rubrics, and model-level scores tend to concentrate within a narrow range. In contrast, our benchmark is constructed to natively support multiple evaluation protocols, including absolute scoring and pairwise comparison-based evaluation. This design provides more stable and discriminative metrics for evaluating whether models truly incorporate user preferences.

Personalized Dialogue System Personalization has long been a focal point in dialogue system research, typically categorized into: 1) role-playing, where models imitate a specific persona (Cheng et al., 2024), and 2) user-alignment, where models adapt to the user's personality or preferences (Yuan et al., 2024; Kwok et al., 2024). However, these systems are predominantly built on human-human conversation datasets (Zhang et al., 2018; Joko et al., 2024), which exhibit a fundamental mismatch with human-LLM interaction dynamics. In human-human dialogue, both participants are relatively symmetric conversational agents; in contrast, human-LLM interaction is inherently asymmetric: the user initiates tasks and expresses preferences, while the LLM is expected to execute the task and adapt its response accordingly. **Therefore, preference modeling in this setting is not merely about imitating a persona or maintaining a dialogue style, but about inferring task-relevant preferences that help the model produce more useful responses.** Our dataset is designed around this asymmetric human-LLM paradigm, providing

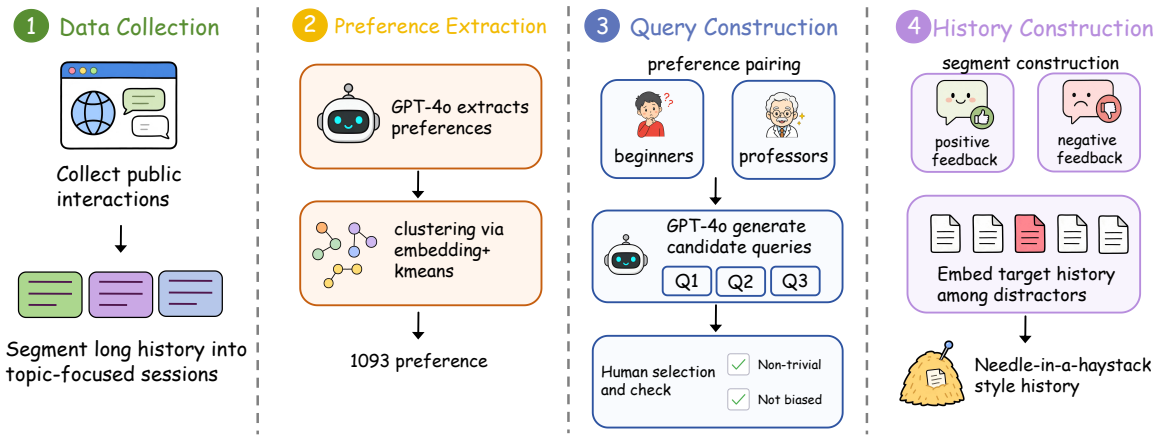

Figure 2: (1) We collected and chunked real interaction history. (2) We extracted the preferences from the chunks, details are provided in Section 3.3.1. (3) We paired the preferences and obtained the query, details are provided in Section 3.3.2. (4) We alternately concatenate the pre-defined templates to construct the interaction history, details are provided in Section 3.3.3.

a more realistic basis for evaluating personalized LLM assistants.

3 InterPref Construction

In this section, we first simply introduce the formulation of DIPA, and then present the construction of InterPref.

3.1 Task Formulation

The goal of **DIPA** is to enable an LLM to adapt its responses to a user's latent preferences directly from interaction history. Here, we define a user's preferences as any information in their past interactions that can help the model generate responses better aligned with their needs, including linguistic style, domain expertise, and specific constraints.

Let $h_u = (q_{u,i}, r_{u,i})_{i=1}^{n}$ denote the user u's history, implicitly encoding preferences $\mathcal{P}u$. Given a new query $qn+1$, the model generates a personalized response:

$$r_{u,n+1} = \text{LLM}(h_u, q_{n+1}), \quad (1)$$

ensuring that users with different histories receive responses aligned with their respective preferences. After each turn, the history is updated as $h_u \leftarrow h_u \cup (q_{n+1}, r_{u,n+1})$.

3.2 Data Collection

We developed a data-collection website that lets the public interact with ChatGPT and record conversations. We collect long-term interaction histories, which may span varied times and topics reflecting users' real-world needs. GPT-4o then segments each history into topic-specific sessions, which serve as the basic units for our synthesis pipeline.

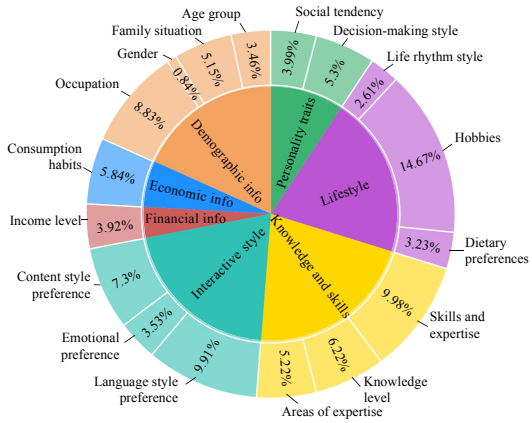

Figure 3: Preferences distribution of our collect dataset, including seven main categories, such as "Lifestyle".

3.3 Data Construction

We construct our dataset to both preserve privacy and enrich preference signals. Original user interactions often contain sparse preference information, so we extract user preferences, generate preference-related queries, and reconstruct interaction histories to form evaluation tasks that focus on user preferences. Each task is a triplet: a preference pair, a history pair, and a preference-related query.

3.3.1 Preference Extraction

We utilised GPT-4o to extract user preferences from each dialogue session. To classify the preference topics, we performed clustering using the K-means algorithm based on embeddings generated

by the `text-embedding-ada-002`, and manually assigned a category name to each cluster. The topic composition of our dataset is shown in Figure 3, covering a wide range of real-world scenarios from daily life. Ultimately, we collected $1,093$ different user preferences.

3.3.2 Query Construction

The query q_{n+1} is designed specifically to evaluate whether the model considers the user's preference. To this end, it must be related to the preference but retain sufficient independence, so that the resulting response can either align with the preference or not, depending solely on the preference inferred from dialogue history. By comparing responses across different preferences within the same topic, we can determine whether the model has appropriately integrated the preferences. To construct such queries, we first create **paired preferences** and then back-formulate q_{n+1} based on these pairs.

Preference Pair Generation Since the collected preferences only sparsely cover the true preference space, we use GPT-4o to generate a distinct paired preference p_u^d for each collected preference p_u. This expands the dataset and increases preference diversity, e.g., from "the user likes eating pizza" we can generate "the user likes burgers" or "noodles".

Query Selection and Verification For each pair of preferences p_u and p_u^d, the LLM generates three candidate queries. Human annotators then filter and verify them according to two criteria: (1) the query should not be trivial—for example, "Do you like eating pizza?" is trivial because it trivially elicits the preference without meaningful reasoning; and (2) the query must be relevant to both preferences. Queries that show a clear bias toward either side are discarded.

3.3.3 History Construction

We adopt a behavior-driven strategy for history construction. Rather than relying on unconstrained generation, we design templates that simulate two distinct patterns of user feedback. We find this approach more effective for implicitly embedding target preferences into the conversational context. These patterns emulate real-world interactions where users respond to LLM outputs based on their level of satisfaction.

- Negative Feedback with Correction (Rewrite): The user provides a critique or specific comments on the initial response, prompting the model to regenerate the answer to align with the preference. This simulates a corrective interaction.
- Positive Feedback with Extension (Follow-up): The user expresses interest in the initial response and asks further questions to explore specific details. This simulates an appreciative interaction that naturally sustains the preference.

Metric	Syn.	Collect	Sharegpt
Language	En	En & Zh	En
#Samples	800	700	250
Avg. Turns	20	28.6	45.6
Avg. Tokens/Turn	143.8	122.3	174.8

Table 1: Dataset statistics. "Syn." denotes the synthetic data, and "Collect" denotes the original data we collected.

To better simulate real-world interaction histories in which users may exhibit preferences across multiple domains, we embed the interaction history associated with the ground-truth preference among those of four distractor preferences and concatenate them, resulting in a needle-in-a-haystack–style dialogue history (Hsieh et al., 2024).

3.3.4 Data Quality Verification

We verified the quality of the dialogue history from the following two perspectives:

Preference Preservation To verify whether the synthetic conversations preserve the intended preferences, we ask Llama3.1-70B to re-extract preferences from the generated histories. The construct–extract consistency reaches 94%, indicating that most target preferences are successfully retained. This suggests that, when no distractor histories are introduced, and the model can focus on the target dialogue, a strong LLM can reliably recover the underlying preferences. Therefore, the benchmark's upper bound is not limited by noise in the synthetic histories.

Lexical Diversity Lexical Diversity. We further examine whether the constructed conversations achieve lexical diversity comparable to real conversations. We compute self-BLEU scores, where lower values indicate higher diversity. Our dataset obtains a self-BLEU (Zhu et al., 2018) score of 0.477, lower than the average score of 0.505 across comparable datasets, with details provided in Appendix A. This suggests that our constructed conversations maintain a moderate level of lexical diversity.

3.3.5 Statistical Analysis

We present the key statistics of the original and synthetic datasets in Table 1, comparing with the widely used Sharegpt (shareAI, 2023) dataset.

4 Experiments

We first introduce the metrics, then we evaluate widely used LLMs on our benchmark.

4.1 Evaluation Metrics

4.1.1 Absolute Judge Metrics

Given a user query q_u, the model's response r_u, and the ground-truth preference text p_u, which serves as the gold standard for alignment, the absolute evaluation score is formally defined as:

$$\text{score} = \text{eval}(r_u | q_u, p_u,) \quad (2)$$

where eval$(\cdot)$denotes the judge (LLM or human) that maps r_u to a numerical value. Specifically, we evaluate the following two aspects: (1) **task completion score** (TCS): Evaluating user preference alignment presupposes that the model has completed the user's fundamental task. Based on this principle, we employed absolute evaluation with a 1-5 scoring scale to assess the overall quality of task completion. (2) **preference alignment score** (PAS): We also report the absolute preference alignment scores, scoring from 1-5. The scores are evaluated by distinct evaluators (GPT-4o, Llama3.1-70B, Qwen2.5-72B, human). The prompt and detailed scores of each evaluator for every model are in Appendix I.

4.1.2 Comparitive Judge Metrics

Evaluating the alignment between responses and user preferences is inherently complex due to the subjective nature of preference interpretation. Our pilot studies reveal two major issues with the absolute scoring of individual responses. 1) **Different evaluators adopt different scoring standards.** For example, Llama3.1-70B tends to assign high scores, with most scores ranging between 4 and 5, whereas Qwen2.5-72B is relatively strict, rarely assigning a score of 5 and instead tending to give lower scores. This leads to low consistency among different evaluators, with a Fleiss' Kappa coefficient of only 0.236. 2) **Absolute scores have limited discriminative power.** Across different models, the average PAS scores are mostly concentrated within the range of 3.5 to 4.5. Therefore, we shift from absolute scoring to a paired comparative evaluation framework, which introduces two key metrics:

Pass Rate The core hypothesis of this metric is that: If the evaluator can reliably identify which response targets which preference, we can conclude that the LLM possesses alignment capability. We formalize this metric as follows:

$$\text{pass} = \sum_{i=1}^{N} \mathbb{I}\Big(\text{eval}(r \rightarrow p \text{ and } r^d \rightarrow p^d)\Big), \quad (3)$$

$r \rightarrow p$ indicates that response r is correctly matched to preference p by the evaluator. The proportion of successful matches by the evaluator across the entire dataset is referred to as the pass rate. We verify that the biases rarely affect the pass rate, and the consistency between the capable LLM evaluator and humans is up to 83%, detailed in Appendix C.

Win Rate Furthermore, even if both models can generate differentiable responses, is there a difference in the quality of their alignment with the preferences? To investigate this, we conduct a metric where annotators compare responses from the two models to select the response pair that better reflects the given preferences. The win rate score of model i of model j is calculated as:

$$\mathbf{w}(i,j) = \sum_{i=1}^{N} \mathbb{I}\Big(\text{eval}\left((r_i, r_i^d) \succ (r_j, r_j^d)\right)\Big), \quad (4)$$

Here, r_i denotes the response of model i. This metric introduces **model-wise** comparison. We use the Bradley-Terry (Bradley and Terry, 1952) model to aggregate the win rate matrix, which measures the relative abilities of two models, into an absolute measure of a single model's ability. We set the win rate as a fully human-evaluated metric to avoid potential self-preference bias that may arise from model-based evaluation. We collaborate with a professional annotation company to ensure that all annotators are well-trained, well-educated, and fairly compensated. Detailed annotation documents are provided in Appendix F.

4.2 Main Experiments

Settings For the pass rate, we use Llama3.1-70B as the evaluator. To reduce the potential influence of positional bias of the LLM evaluator, we

Models	TCS↑	PAS↑	Win Rate				BT-coe↑
			ag.GPT-4o	ag.Llama3.1-70B	ag.Llama3.1-8B	ag.Qwen2.5-7B	
GPT-4o	4.10	3.86	——	0.51	0.65	0.75	0.983
Llama3.1-70B	3.55	3.68	0.48	——	0.58	0.77	0.949
Llama3.1-8B	3.10	3.48	0.34	0.41	——	0.60	0.424
Qwen2.5-7B	3.82	2.84	0.23	0.22	0.39	——	-0.356

Table 2: Win rate matrix of four models. Every model in a row is compared with the remaining three models in the column to calculate the win rate. (ag. represents "against"). All data points are calculated independently. "TCS" denotes the task completion score. "PAS" denotes the preferences alignment score.

Model	TCS	PAS	Pass Rate/%↑
Qwen2.5-7B	4.38	3.46	20.5
Qwen2.5-14B	4.33	3.69	28.0
Qwen2.5-32B	4.42	3.6	22.0
Qwen2.5-72B	**4.45**	3.97	46.5
Qwen3-8B-no think	4.10	3.88	53.5
Qwen3-8B-think	4.26	4.02	53.8
Qwen3-32B-no think	4.35	4.13	59.5
Qwen3-32B-think	4.32	4.23	62.5
Mistral-7B	3.80	3.38	26.0
Mistral-Large-2407	4.22	4.07	**66.0**
Llama3.1-8B	3.42	3.38	52.5
Llama3-70B	4.05	3.77	56.0
Llama3.1-70B	3.97	3.82	57.0
Llama3.3-70B	4.15	3.83	56.5
Llama3.1-405B	4.15	3.98	63.0
Gemini-1.5-flash	4.25	3.98	53.0
Gemini-1.5-pro	4.17	4.17	61.5
Gemini-2.0-flash	4.39	4.14	60.5
GPT-3.5-turbo-1106	4.29	3.95	51.0
GPT-4-turbo	4.27	3.96	54.5
GPT-4o-mini	4.39	3.99	56.5
GPT-4o	4.36	4.03	56.5
GPT-o1-mini	4.42	**4.25**	62.5
DeepSeek-V3	4.33	3.97	50.5
DeepSeek-R1	4.36	4.07	60.5
Human	3.49	3.43	55.0

Table 3: The performance of different model series. The evaluator is Llama3.1-70B, Qwen2.5-72B, and GPT-4o.

swapped the order of preferences and responses in the evaluator's prompt. We ran the evaluation four times with different orders. The evaluator is considered correct only if it identifies the right match in all four cases. For the win rate, we selected four models of varying sizes and series to construct a small-scale chatbot arena (Chiang et al., 2024) experiment. We sample 200 pairs of interaction histories from the entire dataset as the test set.

Results All the absolute judge metrics and pass rate are shown in Table 3. The win rate matrix is shown in Table 2. Our findings are:

(1) The model's capability on the DIPA task is significantly different from its traditional reasoning capability. Model series that perform strongly in reasoning tasks, such as Qwen2.5 (Qwen, 2024), tend to perform poorly on DIPA. Similarly, within a model series, for example, the reasoning ability on code and math of Llama3.3-70B is significantly stronger than that of Llama3.1-70B (Meta, 2024), yet its capability on DIPA does not correspondingly improve; the same relationship is also found in GPT-4o and GPT-4o-mini. We speculate that the performance might be more profoundly related to the alignment techniques the model employs, and thus, the capabilities within the same series are more similar.

(2) The ability to engage in slow thinking (Min et al., 2024), or the capability to generate long chain-of-thought style responses, significantly impacts the model's performance. The slow-thinking models, such as GPT-o1-mini and DeepSeek-R1, show significant improvements compared to other models such as GPT-4o-mini and DeepSeek-V3. Furthermore, for the Qwen3 series, where we can manually toggle the slow-thinking mode on or off, the Pass Rate and PAS metrics show the same result: **thinking outperforms no-thinking.** We believe this is because preference alignment tasks are inherently multi-hop. Specifically, the model needs to perform implicit reasoning in the form of $h_u \rightarrow p_u \rightarrow r_{u,n+1}$. Slow-thinking models are naturally better at this type of multi-hop reasoning.

(3) Compared to LLMs, humans are less capable of serving as language assistants that account for complex user preferences. The human baseline is obtained by averaging the annotations of four annotators. We observe that human performance on this task does not surpass that of state-of-the-art LLMs. This is likely due to two main factors: 1. humans are better suited for processing text with logical continuity, whereas the long, topic-discontinuous dialogues in DIPA pose a greater challenge. 2. Under repetitive and tedious annotation tasks, humans

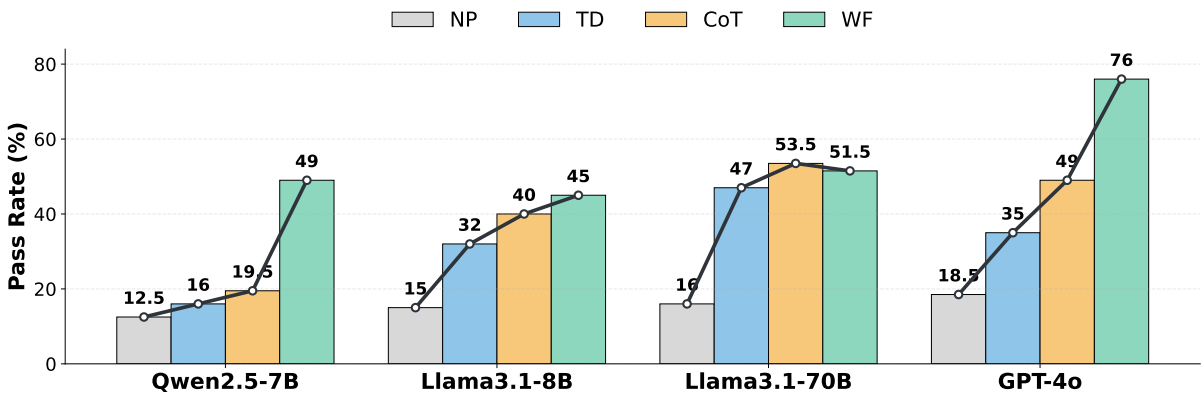

Figure 4: Across model families with varying capabilities, the results on the impact of inference paradigms consistently exhibit the ordering NP < TD < CoT < WF.

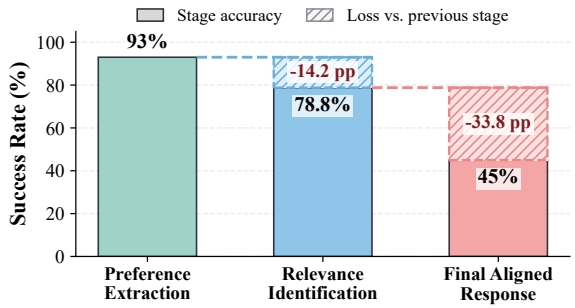

Figure 5: Llama3.1-8b failure mode: The proportion of preferences lost by the model at each stage of the workflow (WF).

tend to write concise responses, and the lower information content increases the likelihood of failing to match user preferences. Specifically, in absolute scoring without comparison, concise responses tend to significantly lower the evaluator's ratings.

4.3 Impact of Inference Paradigms

In this experiment, we further explore the model's performance under different prompting methods, aiming to gain a clear overview of the upper and lower bounds of the model's capabilities.

Settings We selected several inference methods:

- **No-prompting (NP)**: The vanilla LLM without any specific instructions.
- **Task description (TD)**: The LLM is explicitly instructed to act as a preference-aware assistant.
- **Chain-of-Thought**: Based on TD, the model is required to generate a reasoning process before the final answer.
- **Workflow (WF)**: A manually designed three-stage process where the model extracts relevant preferences, identifies the relevance between preference and query, and finally generate the response. This setting simulates an agent-style workflow in which the LLM is invoked multiple times to complete the task. The design details can be found in Appendix E.

Results The result in Figure 4 indicates that:

(1) LLMs possess the capability for preference alignment but lack the spontaneous awareness to apply it. While GPT-4o only achieves an 18.5% pass rate under the NP condition, the significant improvement under the TD condition (e.g., Llama3.1-70B rising from 16% to 47%) indicates that the performance gap stems from task recognition rather than text understanding. This suggests that models have the underlying ability but suffer from "instructional blindness"—they do not prioritize user preferences as key constraints unless explicitly instructed.

(2) Internal integration is the main bottleneck in single-turn inference. While using a structured workflow (WF) gives the best results, it exposes a gap between understanding preferences and integrating them into a final response. In other words, the challenge of DIPA is not in recognizing the user's preferences, but in executing the alignment while producing high-quality text in a single model call. As shown in Figure 5, the model performs very well on intermediate steps—extracting preferences (93% success) and evaluating their relevance (78.8% success)—yet its accuracy drops to 45% when combining these insights into a fully aligned response. This indicates that generating preference-aligned text in one step is the most difficult task. Similar to how humans refine complex ideas through multiple drafts, LLMs benefit from breaking the task into structured sub-steps.

Model	Train Conf.	Data	PAS	Pass Rate↑
Llama3.1-8B	vanilla	Syn.	3.46	20.4
	Distilled	Syn.	3.82	65.0
	vanilla	Real	2.87	12.5
	Distilled	Real	3.28	33.9
Qwen2.5-7B	vanilla	Syn.	3.60	17.0
	Distilled	Syn.	3.84	65.5
	vanilla	Real	2.98	10.4
	Distilled	Real	3.36	33.3

Table 4: Cross-dataset performance comparison. "Syn." denotes the synthetic test set.

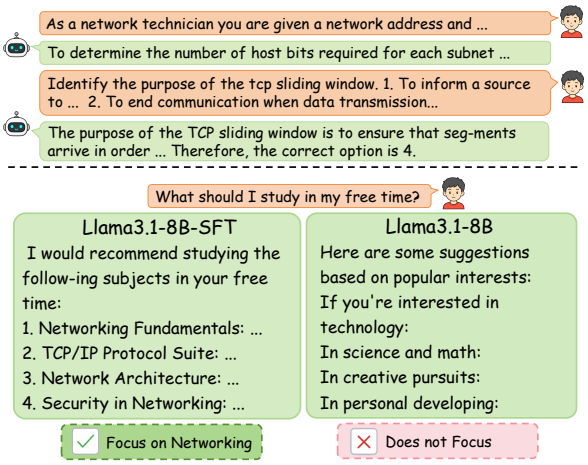

Figure 6: A case study of real-world interaction. Llama3.1-8B-SFT denotes the trained model.

4.4 Capability Distillation

Beyond serving as an evaluation benchmark, our framework can also be used for data distillation to enhance the model's inherent ability.

Settings We use WF-based teacher models to generate responses for 500 held-out samples, while student models are evaluated under the NP paradigm. Responses that successfully incorporate the target preferences are selected via Pass Rate-based rejection sampling, forming a high-quality SFT dataset. We fine-tune Qwen2.5-7B and Llama3.1-8B on this dataset, with training details in Appendix G.

To verify that synthetic fine-tuning effectively generalized to real-world scenarios, we further evaluate generalization on a small-scale, confidential real-world benchmark constructed by manually pairing authentic dialogues. The results on the main dataset and confidential dataset are in Table 4.

Results The results indicate that:

(1) Fine-tuning effectively internalizes complex reasoning workflows. In the synthetic test set, the fine-tuned Llama3.1-8B and Qwen2.5-7B achieved pass rates of 65.0% and 65.5% respectively—a multi-fold increase over their vanilla counterparts. This suggests that the models have successfully transitioned from prompt-dependent alignment to autonomous preference following, effectively distilling the teacher model's WF (Workflow) capability into their own NP (No-Prompting) weights.

(2) The capability enhancement demonstrates strong generalizability. We observed a consistent twofold improvement in the real dataset. The fact that performance gains persist across an unseen dataset confirms that the model is learning the underlying logic of preference alignment rather than merely memorizing synthetic patterns. While the magnitude of the gain is slightly lower than that observed on synthetic data, this is likely due to the distribution shift between structured training templates and more complex real-world user–LLM interactions. Nevertheless, the persistent upward trend highlights the robustness of our distillation procedure across different data distributions.

4.5 Case Study

We further demonstrate the results of the training through detailed case analysis. We selected several typical query types, including planning, creative writing, and recommendations. A successful case of recommendation is demonstrated in Figure 6, and more cases are in Appendix. In Figure 6, the user asked numerous questions about computer networks previously. When consulting the LLM about what to study, the trained LLM inferred that the user is interested in networking and technology-related topics and provided an appropriate response. The original LLM, however, behaves as if the history does not exist, showing that it does not focus on or understand the user's preferences. Beyond the successful cases, we also present a set of failure cases to provide a more comprehensive view of model behavior, detailed in Appendix H.

5 Conclusion

In this work, we introduce InterPref to systematically evaluate inference-time preference alignment in LLMs. Our results show that current models still struggle to incorporate complex user preferences, while structured workflows, slow-thinking, and data distillation can substantially improve preference-aware performance.

Limitation

Lack of immediate user feedback A clear but insurmountable issue is that we don't have the results of immediate real user feedback. Perhaps only companies with large-scale online services, like OpenAI or DeepSeek, can access immediate user feedback, making the research in this topic even more solid. But regardless, we have proposed this problem and utilized as many resources as possible to conduct our research, hoping to provide some insights to the community.

References

Ralph Allan Bradley and Milton E. Terry. 1952. Rank analysis of incomplete block designs: The method of paired comparisons. *Biometrika*, 39(3-4):324–345.

Chuanqi Cheng, Quan Tu, Wei Wu, Shuo Shang, Cunli Mao, Zhengtao Yu, and Rui Yan. 2024. "in dialogues we learn": Towards personalized dialogue without pre-defined profiles through in-dialogue learning. *Preprint*, arXiv:2403.03102.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. Chatbot arena: An open platform for evaluating llms by human preference. *Preprint*, arXiv:2403.04132.

Hyundong Cho, Karishma Sharma, Nicolaas Jedema, Leonardo Ribeiro, Alessandro Moschitti, Ravi Krishnan, and Jonathan May. 2025. Tuning-free personalized alignment via trial-error-explain in-context learning.

Wanwei He, Min Yang, Rui Yan, Chengming Li, Ying Shen, and Ruifeng Xu. 2020. Amalgamating knowledge from two teachers for task-oriented dialogue system with adversarial training. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 3498–3507.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. Ruler: What's the real context size of your long-context language models? *Preprint*, arXiv:2404.06654.

Hideaki Joko, Shubham Chatterjee, Andrew Ramsay, Arjen P. de Vries, Jeff Dalton, and Faegheh Hasibi. 2024. Doing personal laps: Llm-augmented dialogue construction for personalized multi-session conversational search. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 796–806, New York, NY, USA. Association for Computing Machinery.

Louis Kwok, Michal Bravansky, and Lewis D. Griffin. 2024. Evaluating cultural adaptability of a large language model via simulation of synthetic personas. *Preprint*, arXiv:2408.06929.

Young-Jun Lee, Chae-Gyun Lim, Yunsu Choi, Ji-Hui Lm, and Ho-Jin Choi. 2022. PERSONACHATGEN: Generating personalized dialogues using GPT-3. In *Proceedings of the 1st Workshop on Customized Chat Grounding Persona and Knowledge*, pages 29–48, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Meta. 2024. Llama3.3.

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, and Jiahui Wen. 2024. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *ArXiv*, abs/2412.09413.

Rémi Munos, Michal Valko, Daniele Calandriello, Mohammad Gheshlaghi Azar, Mark Rowland, Daniel Guo, Yunhao Tang, Matthieu Geist, Thomas Mesnard, Côme Fiegel, Andrea Michi, Marco Selvi, Sertan Girgin, Nikola Momchev, Olivier Bachem, Daniel J. Mankowitz, Doina Precup, and Bilal Piot. 2024. Nash learning from human feedback. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Lin Ning, Luyang Liu, Jiaxing Wu, Neo Wu, Devora Berlowitz, Sushant Prakash, Bradley Green, Shawn O'Banion, and Jun Xie. 2024. User-llm: Efficient llm contextualization with user embeddings. *Preprint*, arXiv:2402.13598.

OpenAI. 2024. Memory and new controls for chatgpt.

Divya Patel, Pathik Patel, Ankush Chander, Sourish Dasgupta, and Tanmoy Chakraborty. 2024. Are large language models in-context personalized summarizers? get an iCOPERNICUS test done! In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16820–16842, Miami, Florida, USA. Association for Computational Linguistics.

Lisong Qiu, Juntao Li, Wei Bi, Dongyan Zhao, and Rui Yan. 2019. Are training samples correlated? learning to generate dialogue responses with multiple references. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3826–3835.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang

Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Qwen. 2024. The official repo of qwen.

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.

shareAI. 2023. Sharegpt-chinese-english-90k bilingual human-machine qa dataset. https://huggingface.co/datasets/shareAI/ShareGPT-Chinese-English-90k.

Yueming Sun and Yi Zhang. 2018. Conversational recommender system. *Preprint*, arXiv:1806.03277.

Hiromi Wakaki, Yuki Mitsufuji, Yoshinori Maeda, Yukiko Nishimura, Silin Gao, Mengjie Zhao, Keiichi Yamada, and Antoine Bosselut. 2024. Comperdial: Commonsense persona-grounded dialogue dataset and benchmark. *Preprint*, arXiv:2406.11228.

Ruifeng Yuan, Shichao Sun, Yongqi Li, Zili Wang, Ziqiang Cao, and Wenjie Li. 2024. Personalized large language model assistant with evolving conditional memory. *Preprint*, arXiv:2312.17257.

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? *Preprint*, arXiv:1801.07243.

Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. 2025. Do llms recognize your preferences? evaluating personalized preference following in llms. *Preprint*, arXiv:2502.09597.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 1097–1100, New York, NY, USA. Association for Computing Machinery.

Dataset	self-BLEU↓
shareGPT *(real)*	0.361
LAPS (Joko et al., 2024)	0.399
ComperDial (Wakaki et al., 2024)	0.479
PersonaChatGen (Lee et al., 2022)	0.638
InterPref *(ours)*	0.477

Table 5: The diversity of LLM-synthesized personalized dialogue datasets measured by self-BLEU. The tokenizer is cl100k_base. The values represent the average self-BLEU-N scores (N $\in 1, 2, 3, 4$).

Dataset Split	Pass Rate	Avg.Character Length
longer half	53.0	1,388.6
shorter half	59.0	555.7

Table 6: Evaluation of different splits of the response length of GPT-4o

A Interaction Record Quality

One key consideration for an LLM-based synthetic interaction history is the diversity in the patterns of LLM-synthesized dialogue data. Data lacking diversity may have potential distribution biases compared to real-world application scenarios, hence we calculate diversity metric **self-BLEU** (Zhu et al., 2018) for our dataset and some previous work related to personalization. We randomly sampled 100 dialogues, truncating each to 580 tokens (the average history length for our dataset for a single preference) to calculate the self-BLEU. We repeated the above experiment 100 times and took the average of the results, as shown in Table 5. The experimental results show that our dataset surpasses the widely used dataset PersonaChatGen (Lee et al., 2022) and the recently constructed dataset (Wakaki et al., 2024), but does not achieve the same level of diversity as real-world data or work focused on enhancing diversity (Joko et al., 2024).

B Calculation of Bradley-Terry Coefficient

Assuming the BT coefficients of the models constitute a vector ξ, the win rate of model m_1 against model m_2 is given by:

$$P(m_1 \succ m_2) = \frac{e^{\xi_1}}{e^{\xi_1} + e^{\xi_2}} \tag{5}$$

To minimize the error between the actual win rates and the expected win rates, we use the following formulation:

$$\arg\min_{\xi} \sum_{i,j} \mathcal{L}\left(W_{ij}, \frac{e^{\xi_i}}{e^{\xi_i} + e^{\xi_j}}\right) \tag{6}$$

Here, W_{ij} is the element in the win rate matrix at the i-th row and j-th column, representing the actual win rate of model i against model j. We choose cross-entropy loss as $\mathcal{L}$, which means $\mathcal{L}(a, a') = -a\log(a') - (1-a)\log(1-a')$.

We use MATLAB's optimization solver to compute the global optimal solution.

C Metric Robustness

C.1 The bias in the evaluator

Through experiments, we verified that several common LLM biases are not present in our evaluators.

C.1.1 Self-preference bias

We used different evaluators to evaluate the data from different generators, and the results are shown in Table 8. The experimental results show that the pass rates calculated by different evaluators are highly consistent, thus being less affected by self-preference bias.

C.1.2 Length bias

We analyze this problem from two perspectives:

Response length within the same model The longer response of the model does not achieve higher metrics. We grouped GPT-4o's response pairs by length into two equal sets (long half and short half) and used Llama3.1-70B as the evaluator to compute pass rates. The results are shown in Table 6.

Response length across different models Models that generate longer responses do not necessarily achieve higher metrics. We were surprised to find that Qwen2.5-7B produced the longest average responses under the given prompt, followed by GPT-4o, Llama3.1-70B, and finally Llama3.1-8B.

C.2 Consistency between the metrics

The Pearson correlation coefficient between pass rate and BT coefficient is 0.959 (p-value $=$ 0.0414). This indicates that calculating the win rate among a set of models is as effective as calculating the pass rate for a single model.

Prompt of LLM-as-a-Judge

Please rate the following conversation on a scale of 1 to 5 (5 being the best):
You need to consider how well the response aligns with the user's preferences (how well the response complete the user's task in TCS), with the mean score maintained at 2.5.
Use the following format to provide your rating:
score: 1

Figure 7: Prompt of LLM-as-a-Judge

C.3 Consistency between LLM and human

To verify the reliability of LLM evaluators, we sampled 100 responses and distinguished them by humans. The result is that GPT-4o, as an evaluator, achieves an 87% consistency with human judgments, compared to 83% for Llama3.1-70B.

D Details of Confidential Metrics

We show the prompt of LLM evaluators in Figure 7. The detailed evaluation scores from each evaluator for every model are provided in Table 7. Without comparison, evaluators tend to assign scores predominantly in the 2-4 range, rarely giving extreme scores, resulting in minimal differentiation between model ratings.

E Human-designed Workflow

In Section 4.3, we have built a human-designed workflow to enhance the capability of models. Here, we will provide the details. We divide the entire DIPA task into to subtasks. (1) **Preference extraction**: We prompt the model to extract preferences from the interaction history. Among the extracted preferences, only a small subset is relevant to the current query. We ask the model to identify the most relevant preferences. (2) **Response generation**: Based on the preferences, the model generates instructions for the response, such as "Since the user loves drinking coffee, I should increase the emphasis on coffee when recommending drinks." And then, the model integrates the instructions and generates the final response.

F Annotation Document

The pictures displayed in the document are too lengthy to show. In the paper, we only present an overview of our interface. Please do not be puzzled by our descriptions of the images in the text; the actual document does contain the corresponding images. The interface is shown in Figure 8.

F.1 Main Guideline

1. Enter the username and click "Start Annotation" (red box).
2. Below are the responses generated by the two models for the same question from two different users (green box).
3. Below that is the reasoning provided by the AI after comparing the two responses, used to determine whether "Left" or "Right" corresponds to the "current user preference information."

After scrolling down the page, you can see the personal information of the two users.

The two responses above are made considering the personal information of the two users. For example, if the user is a person in their 60s, when creating a personalized exercise plan for this user, it is advisable to prioritize recommending exercises with lower intensity.

The light blue box is the selection box, where annotators choose the model that better takes into account the user's information to make the corresponding response. After making a selection, click "Confirm" to complete one annotation.

F.2 Description of the Specific Tasks

Annotators are required to compare the distinct responses generated by the two models and select the model that has taken the user's personal information into greater consideration, which is essentially the better model.

How can we determine whether a model has considered the user's persona? When a model produces significantly different responses to the same question, and these differences are logically related to the user's persona information, we can infer that the model has considered the persona. Conversely, if the model generates similar responses to the same question, it suggests that the persona information has not been taken into account.

To assist annotators in extracting relevant information from lengthy responses, we employ AI-aided reasoning. The AI will extract pertinent details from the two responses and make its own

Generator	PAS-Evaluator				TCS-Evaluator			
	llama3.1-70B	qwen2.5-72B	gpt-4o	human	llama3.1-70B	qwen2.5-72B	gpt-4o	human
qwen2.5-7B	3.69	3.55	3.14	2.84	4.8	3.975	4.37	3.82
qwen2.5-14B	3.86	3.63	3.52	—	3.96	4.72	4.31	—
qwen2.5-32B	3.81	3.61	3.38	—	3.98	4.92	4.38	—
qwen2.5-72B	4.21	3.90	3.82	—	4.01	4.91	4.43	—
mistral-7B	3.60	3.35	3.18	—	3.69	4.19	3.52	—
mistral-large	4.25	3.89	4.08	—	3.96	4.74	3.97	—
llama3.1-8B	3.56	3.29	3.28	3.48	3.77	3.48	3.02	3.10
llama3-70B	4.00	3.69	3.63	—	3.89	4.47	3.78	—
llama3.1-70B	4.00	3.65	3.81	3.68	4.34	3.87	3.70	3.55
llama3.3-70B	4.10	3.73	3.67	—	3.95	4.69	3.81	—
llama3.1-405B	4.21	3.82	3.92	—	3.96	4.70	3.79	—
gemini1.5-flash	4.17	3.82	3.97	—	3.98	4.82	3.96	—
gemini1.5-pro	4.43	3.93	4.15	—	3.96	4.88	3.98	—
gemini2.0-flash	4.42	3.94	4.07	—	4.00	4.93	4.24	—
gpt-3.5	4.19	3.80	3.85	—	3.94	4.77	4.16	—
gpt-4-turbo	4.21	3.8	3.86	—	3.94	4.73	4.14	—
gpt-4o-mini	4.28	3.83	3.88	—	3.97	4.83	4.22	—
gpt-4o	4.24	3.88	3.98	3.86	4.82	3.97	4.29	4.10
gpt-o1-mini	4.46	4.05	4.24	—	4.01	4.96	4.29	—
deepseek-v3	4.19	3.82	3.91	—	4.84	3.96	4.19	—
deepseek-r1	4.31	3.87	4.03	—	4.88	3.95	4.24	—
human	3.88	3.25	3.15	—	3.75	3.45	3.27	—

Table 7: Detailed score of PAS and TCS.

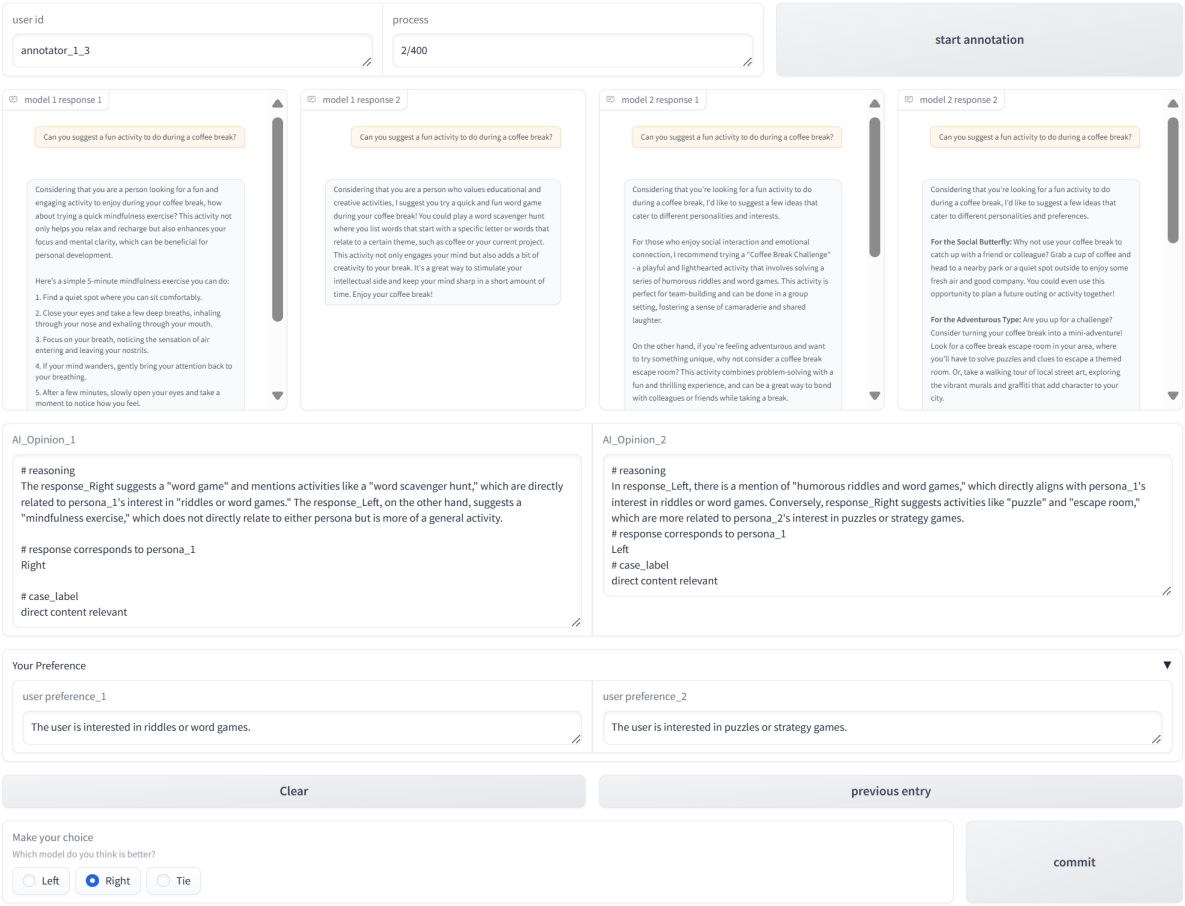

Figure 8: Interface for the annotation.

Generator	Evaluator	
	GPT-4o	Llama3.1-70B
GPT-4o	57.0	56.5
Llama3.1-70B	60.5	57.0
Llama3.1-8B	53.0	52.5

Table 8: Comparison of Pass Rate by different evaluators. Llama3.1-8B is not used as an evaluator due to its weak discriminative capability.

judgment. However, the AI's judgment is often unreliable, and annotators need to review the AI's reasoning and verify it against the responses. If the AI's judgment is flawed, annotators should make their own assessment.

After identifying the relevant information, the final step for annotators is to determine which side has a more significant difference that is related to the user's persona information. They should then select the side with the greater and more reasonable difference. If the differences on both sides are minimal or both are substantial and indistinguishable, the annotator may choose "Tie."

F.3 Specific Examples

The pictures displayed in each example are too lengthy to show.

F.3.1 Example One

First, read the AI's opinion and the user's persona information. Then verify the logic from the response. The AI's logic can be correct in some cases and incorrect in others. For example, the phrase "I hope this message finds you well" appears on one side, and a similar phrase "I hope you're doing well" appears on the other. Therefore, this cannot serve as a basis for determining that one response is more respectful. However, overall, the response on the right does seem somewhat more direct than the one on the left, but the similarity between the two remains quite high. Comparing the two sides, the differences between the two responses from Model 2 on the right seem to be greater. Therefore, the choice is Right.

F.3.2 Example Two

As shown in the AI Opinion, strength training and HIIT, which are high-intensity exercises, appear on the right side and are more suitable for young people around 30 years old. Additionally, although the term "Strength Exercises" appears on the left side, the specific exercises mentioned—squats, wall push-ups, and seated leg raises—are not high-intensity activities and are more appropriate for individuals in their 60s. Therefore, both responses are quite ideal. This time, the differences on the left side are more significant, so the choice is Left.

G More Case Study

In this section, we present a comparison of model performance before and after training on several classic tasks, which helps provide a better understanding of the goals we aim to achieve. In the main text, we provided a case for the recommendation task. Here, we present cases for creative writing, career advice, and planning tasks in Figure 9.

In subfigure (a), the user and the LLM have discussed issues related to TV shows in the interaction history. In the current interaction, the user requests to write a study plan. The Llama3.1-8B, which did not consider the interaction history, merely generated a template involving generic steps. However, Llama3.1-8B-SFT, which considered the user's interest in screenwriting, constructed a targeted study plan. Similar situations also arise in subfigure (b) and subfigure (c), which cover other kinds of tasks.

In subfigure (d), we listed some typical failure cases. They mainly include:

- Preferences that require dedicated construction of the responses.
- Some preferences require in-depth thinking to extract from history. e.g. age.
- Some preferences require in-depth thinking to extract from history, such as "motivational" or "critical".

These complex preferences are difficult to enhance through simple SFT and await the emergence of new alignment methods.

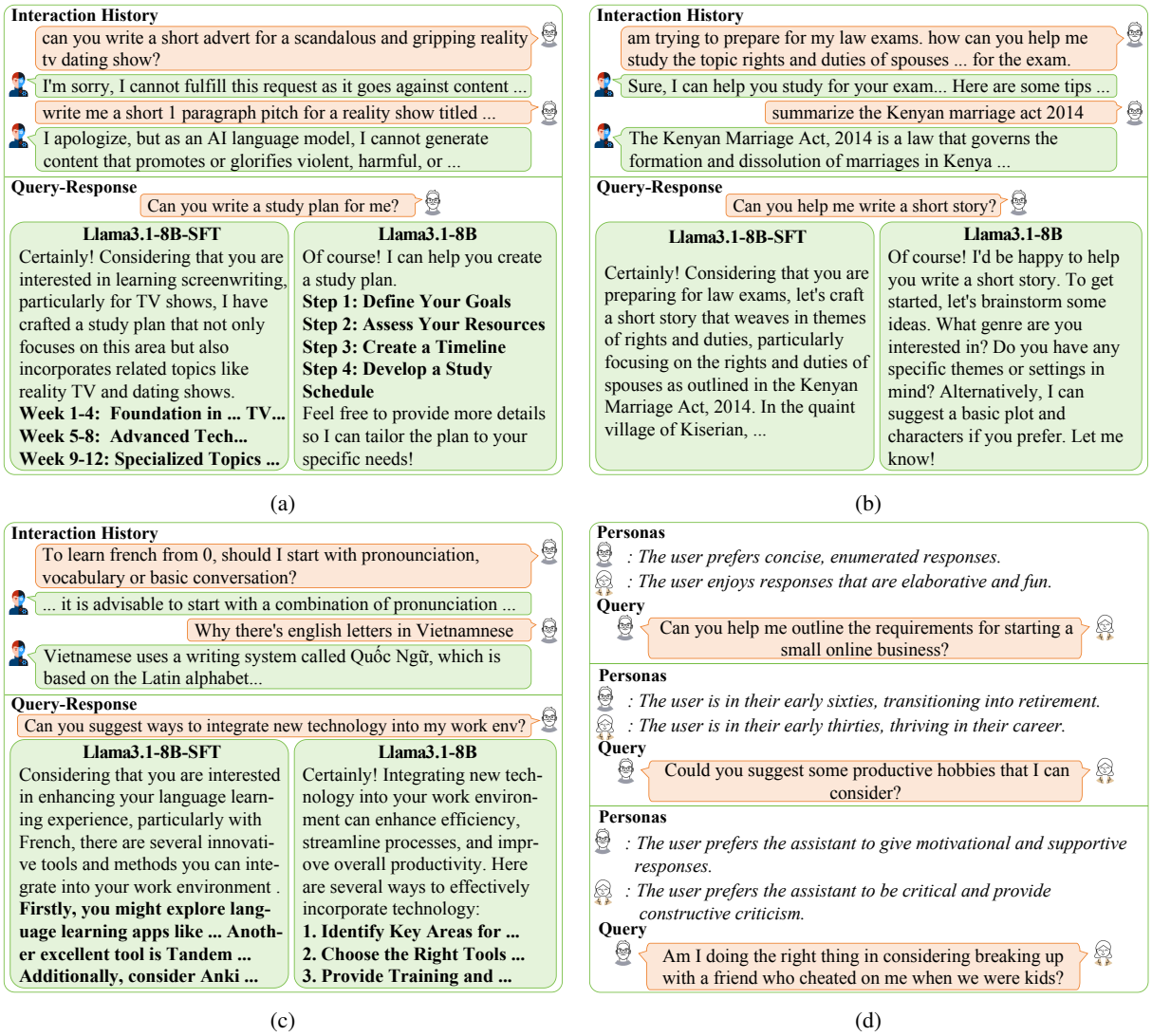

Figure 9: Case study of real-world interactions. (a-c) cases for planning, creative writing, and suggestions. Llama3.1-8B-SFT considered more user preference compared to the original Llama3.1-8B (d) Some preferences and query of failure cases.

H Prompt

Here, we provide the prompt used during the data construction process and evaluations.

H.1 Preference Construction Prompt

The prompt for **preference extraction** is shown in Figure 10.

H.2 Query Construction Prompt

The prompt for **Preference Pair Generation** is shown in Figure 11. The prompt for **Query Generation** is shown in Figure 12.

H.3 History Construction Prompt

The history construction includes two types of logic: **rewrite** and **follow-up**. Here, we list the prompts for both. The prompt for the rewrite is shown in Figure 13 and the prompt for the follow-up is shown in Figure 14.

H.4 Evaluation Prompt

The evaluation prompt for the main experiment is shown in Figure 15. The TD and CoT mentioned in Section 4.3 are only modified for this prompt and are not listed repeatedly.

Preference Extraction

Now your goal is to accomplish topic discovery and user preference extraction tasks from a long dialog history. Let's briefly describe your task:
1) Input: You will be presented with a long, multiturn dialog between a user and an assistant. You can only see the user's queries.
2) Your task:
a) Firstly, you need to partition the whole dialog into multiple chunks by topics: consecutive dialogs about same topic should be put into the same chunk.
- Every dialog within the same chunk is about the same topic and discusses the same matter!!
- The chunk is represented by two [Dialog ID] shown in the dialog history, so the messages between these two [Dialog ID] is a chunk.
- The chunk should begin with a query INDEPENDENT of previous dialog content, which makes the following dialog distinct from previous dialog.
- If a topic has fewer than three dialog turns, do not consider it.
b) Next, for each recommended chunk, generate the following content with English : i) Give a BRIEF topic of this dialog chunk. ('topic')
ii) Give the beginning and end dialog ID, be accurate. ('begin_dialog_id' and 'end_dialog_id')
iii) Extract the user's some key personal information, such as location, job details, interests and hobbies, family background, health status and so on FROM dialogs in this chunk: ('personal_profile')
iv) Extract how formal or casual the assistant's response should be, how long or short responses should generally be, and what type of solutions or information the user prefers to receive FROM dialogs in this chunk: ('response_format')
Output your response in this format.

```
"'json
{"chunks":[
{
"begin_dialog_id": xxx,
"end_dialog_id": xxx,
"topic": "here is the topic of this chunk",
"personal_profile": ["xxx","xxx",...],
"response_format": ["xxx","xxx",...]
}, { "begin_dialog_id": xxx,
"end_dialog_id": xxx,
"topic": "here is the topic of this chunk",
"personal_profile": ["xxx","xxx",...],
"response_format": ["xxx","xxx",...]
}
]}
"'
```

Here is the dialog history: {dialog_formatted}
Now, please understand the examples and give your response to the task instruction. Remember, only output "'json"'!!
Furthermore, please make sure to think carefully.
- When providing chunks, please ensure that every dialog within the same chunk is about the same topic and discusses the same matter.

Figure 10: Preference Extraction

Query Generation

You will be given personal information, and your task is to modify it into the same field but with different characteristics from the current one, you can focus on different aspects of the persona, and you should output 2-3 different modified personas. The differences from the original version can range from small to large, meaning that the first output is relatively close to the original version, while the last output shows a significant difference from it.
Important requirement:
1. Avoid just using the input negation without adding new information,for example: for the entry "The user is interested in ...", don't modify it like "The user in not interested in ...".
2. Double negatives are forbidden because they preserve the original meaning
3. Remember to output "'json and "' in your answer! Input:
-personal information
Output:
-reversed personal information
[Begin Example 1]
input personal information: { "persona":"The user regularly suffers from stomach pain." } output:
"'json { "persona":["The user regularly suffers from headache.", "The user has diabetes", "The user is very healthy and strong"] } "'
[End example 1]
[Begin example 2]
input personal information:
{ "persona":"The user likes to drink coffee" }
output:
"'json { "persona":["The user likes to drink milk","The user likes to drink russian soup","The user likes to eat chocolate"] } "'
[End example 2]
[Begin Input]
{persona}
[End Input]
Now give your output:

Figure 11: Prompt of preference pair generation

Preference Pair Generation

Your task is to generate three queries that simulate a scenario where an AI chatbot user requires assistance from the chatbot, based on the personal information and the interaction history. Your tasks should meet the following requirements. Requirements:
-The query should be based on specific needs in life or work.
-You will be provided with two different personal information, and your queries should be focused on the difference between them. This means that the person with different personal information will do your task differently, and your task should maximize this difference.
-Do not mention user preferences directly in the task. Here is an example to demonstrate this:
[Begin Example]
Personal information 1:
{"persona": "The user is familiar with English grammar and vocabulary exercises."}
personal information 2:
{"persona": "The user is familiar with Spanish grammar and vocabulary exercises"}
Bad answer:
“‘json ["Create a daily practice plan for improving English vocabulary.", "Help me write an email in Spanish for a job application.","Suggest learning resources for mastering English grammar."] “‘
Each of these queries directly refers to a user preference ,which is unbalanced.
Good answer:
["Create a daily practice plan for improving language ability.", "Help me write an email in my familiar language for a job application.","Suggest learning resources for mastering the language I’m learning."]
[End Example]
Input:
-two different personal information
-Dialog history Output:
-queries
[Begin personal information 1]
persona
[End personal information 1]
[Begin personal information 2]
diffused_persona
[End personal information 2]
[Begin Dialog history]
dialog_history
[End Dialog history]
remember to output “‘json and “‘ like this:
“‘json
["query1","query2","query3"]
“‘
Now give your question:

Figure 12: Prompt of preference pair generation

Rewrite interaction history generation

Your task is to generate a two-round dialogue between a user and a chatbot based on the personal information that provided bellow.
Input:
-personal information
output:
-new dialog
The dialogue you generate should take the following form:
1. The user give a query about some topic.
2. The chatbot responds an answer that does not correspond to the user's personal information.
3. The user is not satisfied with chatbot's answers, and give some instruction to change the answer.
4. The chatbot gives answers that match the user's preference.
Notice:
1. You'll start a whole new conversation, pick a topic and move quickly into a meaningful discussion, avoiding non-content conversations like greetings.
2. The queries you generate should be colloquial and closer to real users. For example, if your persona is "User is familiar with geographic topics", you can say "I am very familiar with what you are talking about, can you tell me something new or in-depth?" Your instructions should be given based on persona, not following examples. If persona is None, you can freely generate dialogues that do not involve preferences, such as solving math problems, writing code, and other similar tasks.
3. The answers you generate should be longer and detailed, which is like an AI chatbot. Here is some examples to make you understand this better:
persona: The user is in his mid-thirties.
BAD one:Can you generate responses more in line with the preferences of a man in his mid-thirties?
GOOD one: I think you should take my age into consideration. I'm in my 30s
persona: The user is knowledgeable about microcontrollers.
BAD one: Can you modify your answer to fit the needs of someone who knows a lot about microcontrollers?
GOOD one: Oh! I know quite a bit about microprocessors, let's talk about them in depth!
4. In at least one conversation, you need to accurately convey your persona in the user queries, and you can change the way you express it to make it more colloquial.
your output should in the following format:
user:
user_query_1
chatbot:
chatbot_answer_1
user:
user_query_2
chatbot:
chatbot_answer_2
[Begin personal information]
{persona} [End personal information]
Now give your output:

Figure 13: Prompt of rewrite interaction history generation

Follow-up interaction history generation

Your task is to generate a two-round dialogue between a user and a chatbot based on the personal information provided bellow.
Input:
-personal information
-dialog history
output:
-new dialog
In the query section, you should act as a persona user using the chatbot, and in the answer section, you should be a chatbot responding to the user's query. The topics of conversation can be diverse, as long as you believe the user might be interested in them.
Notice:
1. You'll start a whole new conversation, pick a topic and move quickly into a meaningful discussion, avoiding non-content conversations like greetings.
2. The queries you generate should be colloquial and closer to real users. For example, if your persona is "User is familiar with geographic topics", you can say "I am very familiar with what you are talking about, can you tell me something new or in-depth?" Your instructions should be given based on persona, not following examples. If persona is None, you can freely generate dialogues that do not involve preferences, such as solving math problems, writing code, and other similar tasks.
3. The answers you generate should be longer and detailed, which is like a AI chatbot.
4. In at least one conversation, you need to accurately convey your persona in the user queries, and you can change the way you express it to make it more colloquial.
your output should in the following format:
user:
user_query_1
chatbot:
chatbot_answer_1
user:
user_query_2
chatbot:
chatbot_answer_2
[Begin personal information]
{persona}
[End personal information]
Now give your output:

Figure 14: Prompt of follow-up dialogue generation

Main experiment evaluation prompt

You are a helpful chatbot. Your task is to generate a response to the query based on the dialog history. You need to carefully consider the user's persona from the dialog history. persona may contain user profiles or behaviors that the user expects the conversation assistant to exhibit (such as reply style, etc.). Please consider the user's persona when responding and generate responses that match the user's preferences.
Instructions you must follow:
1. The user profile embodied in the conversation may be relevant to the response of the current query, but the association may be implicit and require some prior information.
2. Please pay special attention to past user queries, topics previously discussed, and requirements that users have previously posed to the chatbot. These should be the main focus for you to obtain information about user preferences.
3. User queries may contain a certain degree of ambiguity. At this point, you only need to generate content according to the instructions without asking for additional information.
4. Integrate the preference information into your answer content. Sentences like "Considering your preference for..." or "Considering your interest in" are not allowed to appear in the responses. You need to think about how to integrate personal information into specific content.
You must output in the following format:
persona: Here is the user persona you extract from dialog history.
Answer: Here is the answer to the current query considering the user persona.
For example:
query: Can you recommend some restaurants that I might like?
your output:
persona: The user likes chicken rather than beef.
Answer: Considering your interest in chicken, I would like to recommend ...
Here is the true user query:
{query}
Now give your output:

Figure 15: prompt of the main experiment.