Preferences Influence Deeply: Enhancing Interaction with Large Language Models via Inference-time Alignment

Anonymous ACL submission

Abstract

User-LLM interaction history contains rich user preferences, which may guide the model in generating more personalized responses. There has not been a systematic exploration of whether current LLMs can infer and align these preferences automatically, and to what extent they can. To fill this gap, we have conducted this study on the capabilities of the current LLMs. We begin by formalizing the task and introducing the **InterPref** benchmark for evaluation. This benchmark includes: 1) A set of interaction histories that contains different preferences, constructed through real histories we collected from a self-built temporary website. 2) A systematic evaluation tool kit. We tested the performance of over 20 open-sourced and proprietary LLMs across various scenarios, including the bare model, hand-crafted prompts, human-designed workflows, and fine-tuning. Our findings reveal that this task is an overlooked capability in current LLM alignment. Furthermore, by comparing different models and analyzing the failure cases, we provide insights for enhancing model performance in the future. We demonstrate that fine-tuning on InterPref can make LLM consider more preferences. This exploration paves the way for the development of future powerful personalized AI assistants. The project can be accessed at `https://anonymous.4open.science/r/InterPref`.

1 Introduction

Large Language Models (LLMs) have led to the development of LLM-powered personal assistants (Qiu et al., 2019; He et al., 2020). Different users exhibit varying personal preferences when using LLM assistants, spanning across various fields from daily life to professional knowledge. To take advantage of such preferences, some works focus on integrating a prompt-based memory mechanism into LLM, allowing it to maintain user preferences (OpenAI, 2024). Others have explored generating personalized responses using additional context retrieval (Yuan et al., 2024; Ning et al., 2024). However, although the introduction of external mechanisms is immediate and effective, to what extent can the model's inherent capabilities achieve this? Specifically, if we **directly** provide the interaction history to the model as its context, can the model infer the user's preferences and align them automatically?

In this work, we first propose the task of **Direct Inference-time Preference Alignment (DIPA)** that aims to generate personalized responses to the query directly based on the previous history. We construct **InterPref**, a benchmark designed for comparatively evaluating whether the model's responses align with the different user preferences. The core idea of this benchmark is shown Figure 1.

The InterPref is based on realistic multi-turn user-LLM interactions and provides an unbiased evaluation kit to judge the models' capabilities. We first collect a large amount of real User-LLM interactions from a self-built website and synthesize new histories through a delicate pipeline. The synthetic data will be fully open-sourced, while the real data will remain closed-source due to user privacy concerns. We conducted experiments on synthetic data, the real data we collected, and the open-sourced ShareGPT (shareAI, 2023), and demonstrated their consistency. Any organization can contact us if they want to conduct evaluations using our collected real data.

For the evaluation, we found that the traditional method of using LLM to score individual answers (Kwok et al., 2024) is seriously unstable in our task. Therefore, we adopt the robust paired comparative evaluation method (Chiang et al., 2024; Munos et al., 2024). Our experiments show that this method is consistent between evaluators and not affected by length or self-preference

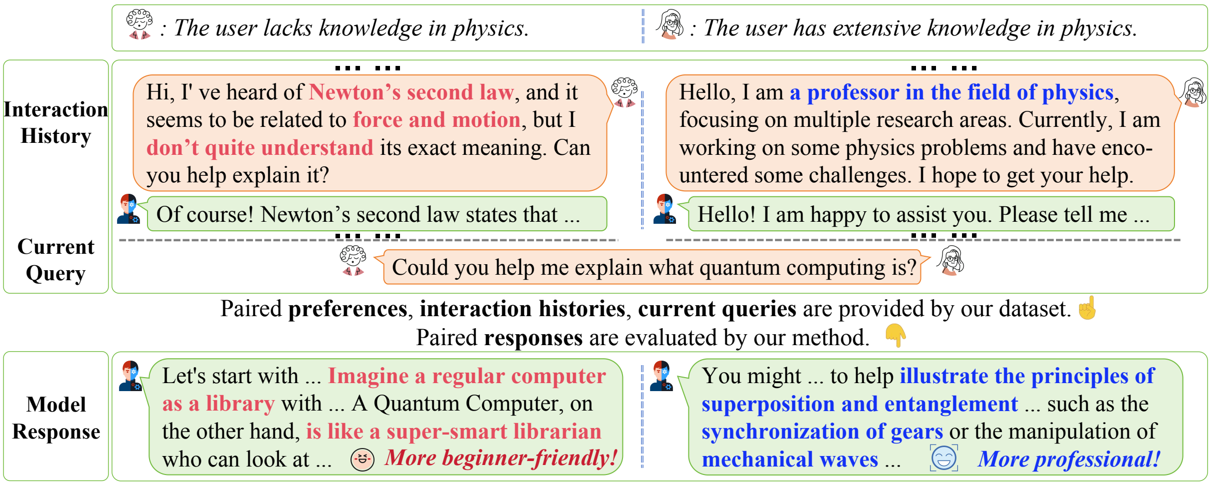

Figure 1: The model understands different preferences embedded within the interaction history and generates responses to the same query that align with distinct user preferences.

bias (Singhal et al., 2024; Panickssery et al., 2024), providing a fair result. Specifically, we designed two metrics:

Pass Rate: This indicator compares the responses of a single model under two different user preferences, shown in Figure 1. It is to measure whether the LLMs **can** generate responses that align with different user preferences or merely output neutral responses regardless of different users.

Win Rate: This indicator measures the sensitivity of different models to the same pair of user preferences. By conducting random pairwise comparisons in a set of models, we use the Bradley-Terry coefficient (Bradley and Terry, 1952) as a measure of the capabilities of a single model.

After evaluating 20 different models and we found that:

- The vanilla LLMs tend to ignore the user's preferences in the history, and only obtain about 20% pass rate on our test set.
- In a hand-crafted prompt, some models automatically exhibit the behavior of gathering preferences from history rather than directly answering the question (similar to slow-thinking), and we have found this to be crucial for the task.
- In the same series, the performance gradually increases with the model size, but a significant gap exists between model series (such as Llama significantly outperforms Qwen). This contrasts with the results observed in reasoning benchmarks such as math, indicating that this capability is independent of those existing ones.
- The current models can only grasp superficial preferences, but cannot grasp preferences that need deeper reasoning, such as the user's preferred conversation style.

We used fine-tuning to make the model inference without hand-crafted prompts, and the fine-tuned Llama3.1-8B achieved a pass rate of about 40.0% on real conversations, showing a significant improvement over the baseline model (20.4%).

2 Related Works

Inference-time Alignment Previous work on inference-time preference alignment has imposed various restrictions on tasks or relied upon some external mechanisms. Some studies have focused on user preferences in specific downstream tasks, such as personalized summarization (Patel et al., 2024; Ao et al., 2021), recommendation (Sun and Zhang, 2018; Zhang et al., 2023), or style imitation (Cho et al., 2025). Some other methods for general tasks presuppose the inclusion of external mechanisms. Some approaches require constructing context before the interaction as a way to cold start (Salemi et al., 2024; Yang et al., 2024). Some necessitate additional user annotations during the interaction (Lau et al., 2024). In contrast, we try to internalize this capability into the model's own parameters, as (Zhao et al., 2025). This aligns more with the trend towards automation and directly reflects the user experience of LLM assistants in their current website form.

Personalized Dialogue System Dialogue systems have been widely studied, and personalization remains a key focus. Existing personalized dialogue systems can generally be categorized into

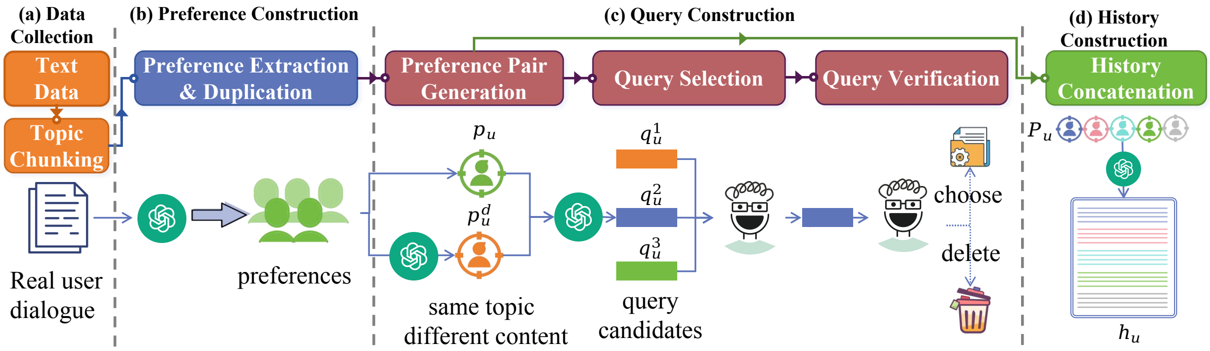

Figure 2: (a) We collected and chunked real interaction history. (b) We extracted the preferences from the chunks, details are provided in Section 3.3.1. (c) We paired the preferences and obtained the query, details are provided in Section 3.3.2. (d) We alternately concatenate the pre-defined templates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

1) systems that require the model to imitate a persona, known as role-playing (Cheng et al., 2024a; Samuel et al., 2024; Cheng et al., 2024b), 2) systems that require the model to align with the user's persona (Yuan et al., 2024; Liu et al., 2021; Kwok et al., 2024). However, these dialogue systems are typically built on datasets based on human-human conversation (Zhang et al., 2018; Wakaki et al., 2024; Joko et al., 2024). The interaction style in these datasets differs significantly from the human-LLM interactions. In our work, we constructed a personalized dataset for the human-LLM interaction style, addressing this critical gap.

3 InterPref Construction

In this section, we first simply introduce the formulation of direct inference-time preference alignment (DIPA), and then present the construction of the synthetic data for InterPref.

3.1 Task Formulation

Given the multi-turn interaction history $h_u = \{(q_{u,i}, r_{u,i}) \mid i = 1, 2, \ldots, n\}$ for the user u. $q_{u,i}$ denotes the i-th query of user u, and $r_{u,i}$ denotes the model's i-th response. Along with the current user query $q_{u,n+1}$, DIPA requires the LLM to capture the user's preferences and generate a user-preferred response $r_{u,n+1}$:

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

Then, r_u and q_u will update h_u, serving as the data source for the model to understand ever-changing preferences.

3.2 Data Collection

As shown in Figure 2(a), we developed a data-collecting website that allows the public to interact

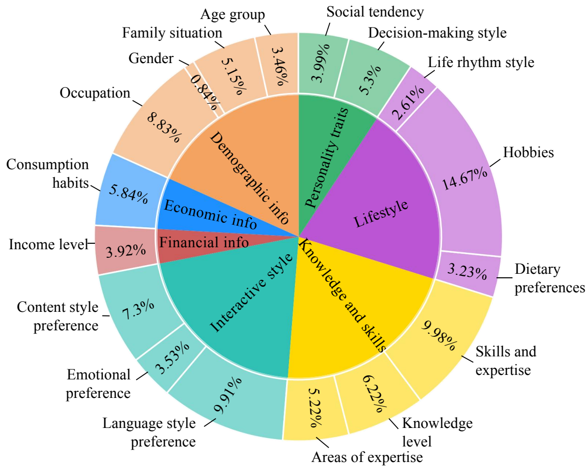

Figure 3: Preferences across multiple domains, including seven main categories, such as "Lifestyle".

with ChatGPT and record their conversations. However, since these conversations contain sensitive personal information, we chose to extract highly abstracted preferences from these conversations and reconstruct new histories. After collecting all the interaction records over a long period (not continuous in time or topic). We first prompt an LLM to chunk the conversation into individual sessions, each focusing on a single topic.

3.3 Data Construction

We introduce the data construction pipeline as follows: (1) Extract preference from real conversations, shown in Section 3.3.1. (2) Construct related queries based on the preferences, shown in Section 3.3.2. (3) Create conversation histories according to the preferences, shown in Section 3.3.3. Our data construction pipeline is illustrated in Figure 2.

3.3.1 Preference Extraction and Duplication

As shown in Figure 2(b), we utilized LLM to extract user preferences from each chunked dialogue. Through observation, we find that there are a large number of similar expressions in user preferences. To eliminate duplicate preferences, we applied a text embedding-based approach and clustered them, assigning a category name to each cluster. Ultimately, we successfully collected $1,093$ different user preferences. The topic composition of these preferences is shown in Figure 3.

3.3.2 Query Construction

In this section, we introduce the construction of the query related to the preference. Considering that the query must be relevant to both preferences used for comparison during the evaluation, we need to pair the preferences first. Then, we use human annotators to select the queries from the model-generated candidates. The overview of this pipeline is shown in Figure 2(c).

Preference Pair Generation Since the preferences are only a sparse sampling of the true preference space, we may not find a suitable pairwise mapping of preferences within the original dataset. Therefore, we employ an LLM-based synthesis approach to generate more preferences for pairing. We denote p_u as the preference extracted from the raw data, and p_u^d as a synthetic paired preference on the same topic, making both relevant to the query. Some examples are shown in Table 1. The paired preference serves as the basis of the query generation and the evaluation.

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 the queries. Annotators can judge the correlation between query and preference based on their common sense, and subsequently filter out some queries whose answers are not open enough.

3.3.3 History Construction

We adopt a template-based method to construct histories with desired preferences, as shown in Figure 2. We observed that template-based construction can better preserve the desired preferences than allowing the model to talk freely. Specifically, we constructed two templates through trials, each containing two rounds of dialogues:

- The user asks a question, and the LLM answers. After that, the user requests the model to re-answer the question through comments, referred to as "rewrite", shown in Figure 4.
- The user asks a question, and the LLM answers. Then, the user continues asking further questions on points of interest, referred to as "follow-up".

Rewrite

user: Hey, I'm planning a trip and looking for some thrilling activities. Got any recommendations for an adrenaline junkie like me?

assistant: Sure! How about a relaxing beach vacation where you can unwind and soak up the sun ...

user: Relaxing isn't really my style. I'm more into action-packed adventures. Can you suggest something that will get my heart racing?

assistant: Absolutely! How about going skydiving or bungee jumping ...

Figure 4: An example of the "rewrite" template.

To better simulate the real-world history, where the conversation is discontinuous in terms of both time and topic. We select five preferences from different categories, concatenate the corresponding histories, and form an interaction history of over 20 turns. We verified the quality of the dialogue history from the following two perspectives:

Preference Preservation Whether the conversations contained the user preference we desired? We required Llama3.1-70B to re-extract preferences from the conversations generated by GPT-4o. The results showed that such a construct-extract operation successfully preserved 94% of the preferences.

Lexical Diversity Can the lexical diversity of constructed conversations be comparable with that of real ones? We calculated the self-BLEU (Zhu et al., 2018) scores (the lower the better) for our dataset (0.477) and other similar datasets (0.505 on average), detailed in Appendix B. The experimental results show that our data maintains a moderate level of lexical diversity among all the datasets.

3.3.4 Construction of Confidential Dataset

We constructed two confidential datasets based on the ShareGPT and the data we have collected. We adopted the same method as described above to construct user preferences and then paired preferences based on GPT-4o. Finally, we have a total of 250 test data points from ShareGPT and 700 data points from our collected data. The data we collected will remain closed-source, but we call on any team to contact us for evaluation.

	Case1	Case2
Preference p_u	The user is in their early sixties	The user enjoys light-hearted and humorous interactions
Preference p_u^d	The user is a young adult	The user prefers serious and formal interactions
Query $q_{u,n+1}$	Can you suggest some enjoyable and fulfilling hobbies?	Can you help me write a speech for my best friend's wedding?

Table 1: Examples of preference pairs and corresponding queries. The topic of preference pairs should be the same.

Model Series	Model	Pass Rate/%↑
Qwen series	Qwen2.5-7B	20.5
	Qwen2.5-14B	28.0
	Qwen2.5-32B	22.0
	Qwen2.5-72B	46.5
Mistral series	Mistral-7B	26.0
	Mistral-Large-2407	**66.0**
Llama series	Llama3.1-8B	52.5
	Llama3-70B	56.0
	Llama3.1-70B	57.0
	Llama3.3-70B	56.5
	Llama3.1-405B	63.0
Gemini series	Gemini-1.5-flash	53.0
	Gemini-1.5-pro	61.5
	Gemini-2.0-flash	60.5
GPT series	GPT-3.5-turbo-1106	51.0
	GPT-4-turbo	54.5
	GPT-4o-mini	56.5
	GPT-4o	56.5
	GPT-o1-mini	62.5
DeepSeek series	DeepSeek-V3	50.5
	DeepSeek-R1	60.5
Human	Human	55.0

Table 2: The pass rate of different model series, using Llama3.1-70B as the evaluator.

4 Experiments

We first introduce the metrics, then we evaluate many widely used LLMs on our benchmark. **We ensure the reproducibility of our results in Section 4.2 and Section 4.3, for they are complete based on the open-sourced synthetic dataset.**

4.1 Evaluation Metrics

Evaluating the alignment between a response and user preferences is inherently challenging due to the subjectivity of preference interpretation. Our pilot experiment shows that the evaluation of a single response is unstable. Specifically, we use GPT-4o, Llama3.1-70B, Qwen2.5-72B, and human annotators to score the degree of alignment on a scale from 1 to 5 on responses generated by GPT-4o. The Fleiss' Kappa (κ) coefficient among all the evaluators is 0.236, indicating that the agreement among evaluators is low. We attribute this inconsistency to the lack of a baseline response for comparison and evaluators' different understandings of the preferences. To address this challenge, we employ a paired comparative evaluation framework (rather than absolute scoring), which improves robustness through relative judgments.

Our comparative evaluation framework introduces two key metrics:

Pass Rate This metric answers one question: Can the LLM generate a response that aligns with different preferences? 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:

$$\text{pass} = \sum_{i=1}^{N} \mathbb{I}\Big(\text{eval}(r \to p \text{ and } r^d \to p^d)\Big), \quad (2)$$

$r \to p$ 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 conducted experiments in Appendix D.1 to verify that the biases rarely affect the pass rate**, and the consistency between the capable LLM evaluator and humans is up to 83%.

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:

$$\mathbf{w}(i,j) = \sum_{i=1}^{N} \mathbb{I}\Big(\text{eval}\left((r_i, r_i^d) \succ (r_j, r_j^d)\right)\Big), \quad (3)$$

Models	Win Rate				BT-coe↑
	ag.GPT-4o	ag.Llama3.1-70B	ag.Llama3.1-8B	ag.Qwen2.5-7B	
GPT-4o	—	0.51	0.65	0.75	0.983
Llama3.1-70B	0.48	—	0.58	0.77	0.949
Llama3.1-8B	0.34	0.41	—	0.60	0.424
Qwen2.5-7B	0.23	0.22	0.39	—	-0.356

Table 3: 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. Due to the annotation including a draw option, the sum of the win rates between the models is slightly less than 1.

Here, r_i denotes the response generated by model i. This metric introduces both **preference-wise** comparison and **model-wise** comparison. We ask annotators to mimic real users in making choices based on their own observations and perspectives. We collaborated with an annotation company to ensure that all annotators were well-educated and fairly compensated. We present the annotation documents in Appendix F. 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.

4.2 Main Experiments

Settings **All models would generate responses based on a hand-crafted prompt.** This is because our pilot experiments show that LLMs have difficulty completing this task without a prompt, and Section 4.3 will discuss this. The prompt is shown in Appendix H. 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 swapped the order of preferences and responses and performed the evaluation four times. The evaluator makes a correct distinction only if it successfully identifies the correct correspondence in all different orders (this also means that the baseline for random guessing is 6.25%.). 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 pass rates are shown in Table 2, and the win rate matrix is shown in Table 3. The Pearson Coefficient between pass rate and win rate is 0.959 (p-value=0.0414). Considering their completely different calculation methods and evaluators, they corroborate each other's validity. Our findings are as follows:

(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, 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, 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, or the capability to generate chain-of-thought (CoT) style responses, significantly impacts the model's performance.** The models in the Llama series perform well on the benchmark, with Llama3.1-70B achieving a pass rate and a BT coefficient close to GPT-4o. A possible reason we found is the response patterns of this series of models. The Llama series models often use phrases like "*Given your (interest/situation/background)*" in their responses. Llama3.1-70B included such prefixes 108 times in its responses, while Llama3.1-8B used them 64 times, and GPT-4o only 17 times. And this emergent CoT-style prefix significantly impacts the model's performance. An interesting comparative result is that: When we explicitly instructed the GPT-4o to use such prefixes (nothing changes other than adding a few tokens in the prompt), its pass rate increased to 69.5%. This may be because such models are capable of recognizing that a summary of preferences from the interaction history is needed. This leads us to believe that the ability for slow thinking is essential in DIPA. Another evidence is that 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.

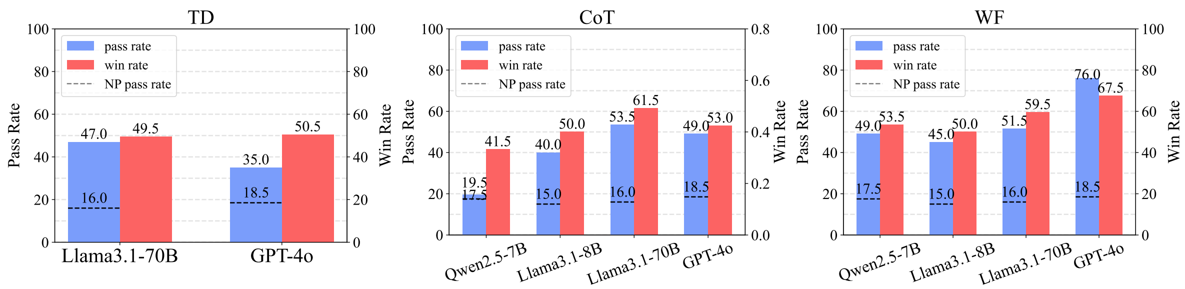

Figure 5: Result of the impact of Inference Paradigms. (left) Performance of models in TD condition. The win rates are calculated by comparing each other. (middle) The performance of models in CoT condition, using Llama3.1-8B as a baseline to get the win rate. (right) Models were performed in WF conditions using Llama3.1-8B as a baseline.

4.3 Impact of Inference Paradigms

In the previous experiment, we mentioned that the models have difficulty completing the task without a prompt. 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 common methods for inference:

- **No-prompting (NP)**: The vanilla LLM.
- **Task description (TD)**: We incorporated a task description in the prompt.
- **Chain-of-Thought (CoT)**: Based on the TD, we further require the LLM to output a chain-of-thought reasoning process.
- **Workflow (WF)**: We manually designed the reasoning workflow for the LLM by decomposing the task into two sub-tasks: first, summarizing the relevant preferences, and then generating the response based on those preferences. The design details can be found in the Appendix E.

Results The result is shown in Figure 5. Our findings are as follows:

(1) Under the NP condition, GPT-4o achieved a pass rate of 18.5, while Llama3.1-70B achieved only 16 (the baseline for random guessing is 6.25%.). This indicates that **current LLMs struggle to automatically consider preference**.

(2) Under the TD condition, the model achieved a significant improvement. The pass rate of GPT-4o increased from 18.5 to 35, while the pass rate of Llama3.1-70B increased from 16 to 47. Also, the chain-of-thought instruction further improves the performance. This indicates that the model's failure under the NP condition is not due to insufficient text understanding capability, but rather because **it may not realize the need to use user preferences to improve its responses**.

Train Conf.	Data	Llama3.1-8B	Qwen2.5-7B
vanilla	Syn.	20.5	17.0
vanilla	Sharegpt	20.4	17.0
vanilla	Collect	12.5	10.4
Self Gen.	Syn.	37.5	51.0
GPT-4o	Syn.	65.0	65.5
GPT-4o	Sharegpt	40.0	36.5
GPT-4o	Collect	33.9	33.3

Table 4: Cross-dataset performance comparison of our trained models. "Self Gen." means that the model is trained by self-generated responses. "Syn." denotes the synthetic test set, and "Collect" denotes the test set using the real dialogue we collected.

(3) When we performed inference in the WF setting, models showed another significant improvement. This suggests that these models possess the ability to complete some subtasks, such as extracting preferences, determining the relevance of the preferences to the current query, and incorporating the preferences into the response, but lack the ability to integrate these subtasks within a single call. By carefully examining the subtasks completed by Llama3.1-8B, we found that the model was able to extract preferences in most tests (with a completion rate of 93%), or recognize that the preferences were relevant to the current query (78.8%), but **constructing a response that aligns with the preferences is relatively challenging** (45%). Given that the generation of any high-quality text by humans requires multiple revisions, constructing an aligned response is indeed challenging in a single call.

4.4 Fine-tuning experiments

In this section, we attempt to use fine-tuning to make the model less reliant on prompts. We train with our synthetic data and test on three datasets from different sources. Subsequently, we conducted a case analysis to ensure that the model's responses indeed incorporated more preferences rather than adapting to latent biases.

Settings We uses 535 training samples, resulting in $1,070$ training instances (with preference being independent during training). We adopted the workflow constructed in Section 4.3 to generate training responses. We trained Llama3.1-8B and Qwen2.5-7B on the response generated by GPT-4o or themselves. We use all the default hyperparameters in the trl Python library, with batch size 1.

Results The results shown in Table 4 indicates:

(1) In the synthetic test set, the trained Llama3.1-8B achieved a pass rate of 65.0%, while Qwen2.5-7B achieved a pass rate of 65.5%, several times higher than the untrained models.

(2) In the confidential test set, both Llama3.1-8B and Qwen2.5-7B achieved a twofold improvement.

(3) Comparing the results of the synthetic test set and the real test set, **the trend of capability improvement remains unchanged, but the improvement is lower in the confidential test set.** This is mainly due to the following two distribution shifts: 1. The synthetic data is constructed based on templates, while real queries are more diverse, such as repetitive inquiries, greetings, etc. 2. The real data we collected is a mixture of English and Chinese, while the training data is purely in English.

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 G. 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.

Although we have demonstrated the success of training, the model's performance is still far from satisfactory. LLMs can only handle some superficial preferences, meaning that the relevance of preferences to the current query does not require in-depth thinking. When facing scenarios that require

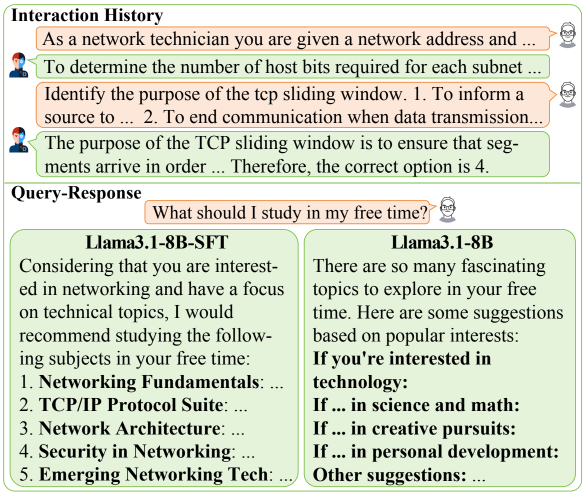

Figure 6: A case study of real-world interaction. Llama3.1-8B-SFT denotes the trained model.

deeper reasoning, the model is unable to provide the desired response.

We summarized several scenarios in which the model still fails to perform well, as follows:

(1) Some preferences require in-depth thinking to extract from history. For example, the LLM did not recognize that the user showed a negative attitude towards life in the conversation.

(2) Some preferences require in-depth thinking the understand the relevance to the current query. For example, LLM did not consider reducing the intensity of physical activities when planning travel for elderly users.

(3) Preferences that require dedicated construction of the responses, such as certain preferences for response styles.

We present examples of preference pairs in Appendix G. **These challenging scenarios is crucial for the usability of personalized LLM assistants, for they reflect many sensitive social issues, such as whether LLM assistants will provide dangerous advice, or guide individuals with negative emotions, and so on.**

5 Conclusion

In this work, we propose a benchmark for direct inference-time preference alignment called InterPref to facilitate a systematic evaluation of the current LLMs. Although the current model's performance does not meet our expectations, our evaluation points the way for future research. The realization of future personal assistants will inevitably require a more meticulous analysis and resolution of these challenges.

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

Xiang Ao, Xiting Wang, Ling Luo, Ying Qiao, Qing He, and Xing Xie. 2021. PENS: A dataset and generic framework for personalized news headline generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 82–92, Online. Association for Computational Linguistics.

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. 2024a. "in dialogues we learn": Towards personalized dialogue without pre-defined profiles through in-dialogue learning. *Preprint*, arXiv:2403.03102.

Yi Cheng, Wenge Liu, Kaishuai Xu, Wenjun Hou, Yi Ouyang, Chak Tou Leong, Xian Wu, and Yefeng Zheng. 2024b. Autopal: Autonomous adaptation to users for personal ai companionship. *Preprint*, arXiv:2406.13960.

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.

Jihyoung Jang, Minseong Boo, and Hyounghun Kim. 2023. Conversation chronicles: Towards diverse temporal and relational dynamics in multi-session conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13584–13606, Singapore. Association for Computational Linguistics.

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.

Allison Lau, Younwoo Choi, Vahid Balazadeh, Keertana Chidambaram, Vasilis Syrgkanis, and Rahul G. Krishnan. 2024. Personalized adaptation via in-context preference learning. *Preprint*, arXiv:2410.14001.

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.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Siyang Liu, Chujie Zheng, Orianna Demasi, Sahand Sabour, Yu Li, Zhou Yu, Yong Jiang, and Minlie Huang. 2021. Towards emotional support dialog systems. *Preprint*, arXiv:2106.01144.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13851–13870, Bangkok, Thailand. Association for Computational Linguistics.

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.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. Llm evaluators recognize and favor their own generations. In *Advances in Neural Information Processing Systems*, volume 37, pages 68772–68802. Curran Associates, Inc.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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.

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.

Vinay Samuel, Henry Peng Zou, Yue Zhou, Shreyas Chaudhari, Ashwin Kalyan, Tanmay Rajpurohit, Ameet Deshpande, Karthik Narasimhan, and Vishvak Murahari. 2024. Personagym: Evaluating persona agents and llms. *Preprint*, arXiv:2407.18416.

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

Prasann Singhal, Tanya Goyal, Jiacheng Xu, and Greg Durrett. 2024. A long way to go: Investigating length correlations in rlhf. *Preprint*, arXiv:2310.03716.

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.

Jing Xu, Arthur Szlam, and Jason Weston. 2022a. Beyond goldfish memory: Long-term open-domain conversation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5180–5197, Dublin, Ireland. Association for Computational Linguistics.

Xinchao Xu, Zhibin Gou, Wenquan Wu, Zheng-Yu Niu, Hua Wu, Haifeng Wang, and Shihang Wang. 2022b. Long time no see! open-domain conversation with long-term persona memory. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2639–2650, Dublin, Ireland. Association for Computational Linguistics.

Rui Yang, Xiaoman Pan, Feng Luo, Shuang Qiu, Han Zhong, Dong Yu, and Jianshu Chen. 2024. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. *International Conference on Machine Learning*.

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.

Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *Preprint*, arXiv:2305.07001.

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.

A More Related Works

In this section, we list some related work because they investigate the ability of LLM that is a part of our required ability.

Memory-based Generation Memory-based tasks require the model to retain memory of previous interactions over time. These tasks include remembering historical events (Xu et al., 2022a; Jang et al., 2023), remembering user personas (Xu et al., 2022b), and memory in agent tasks (Maharana et al., 2024). While these works have focused on expanding memory size, time intervals, and other factors to test the model's memory capacity, the use of memory—an essential aspect for DIPA —is not been adequately explored. Memory-based tasks use keyword-matching algorithms (Papineni et al., 2002; Lin, 2004) as automated metrics because they only employ simple factual question answering to test memory. We believe that simple factual questions are insufficient to demonstrate the impact of memory on interaction, so we include many open-ended tasks in InterPref, where the model needs to consider how the user's preferences impact the complex interaction.

B 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).

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 $\in 1, 2, 3, 4$).

C 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:

$$P(m_1 \succ m_2) = \frac{e^{\xi_1}}{e^{\xi_1} + e^{\xi_2}} \tag{4}$$

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

$$\arg\min_{\xi} \sum_{i,j} \mathcal{L}\left(W_{ij}, \frac{e^{\xi_i}}{e^{\xi_i} + e^{\xi_j}}\right) \tag{5}$$

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.

D Metric Robustness

D.1 The bias in the evaluator

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

D.1.1 Self-preference bias

We used different evaluators to evaluate the data from different generators, and the results are shown in Table 7. The experimental results show that the pass rates calculated by different evaluators are highly consistent, thus being less affected by self-preference bias.

D.1.2 Length bias

We analyze this problem from two perspectives:

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

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

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.

D.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.

D.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.

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 7.

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 pro-

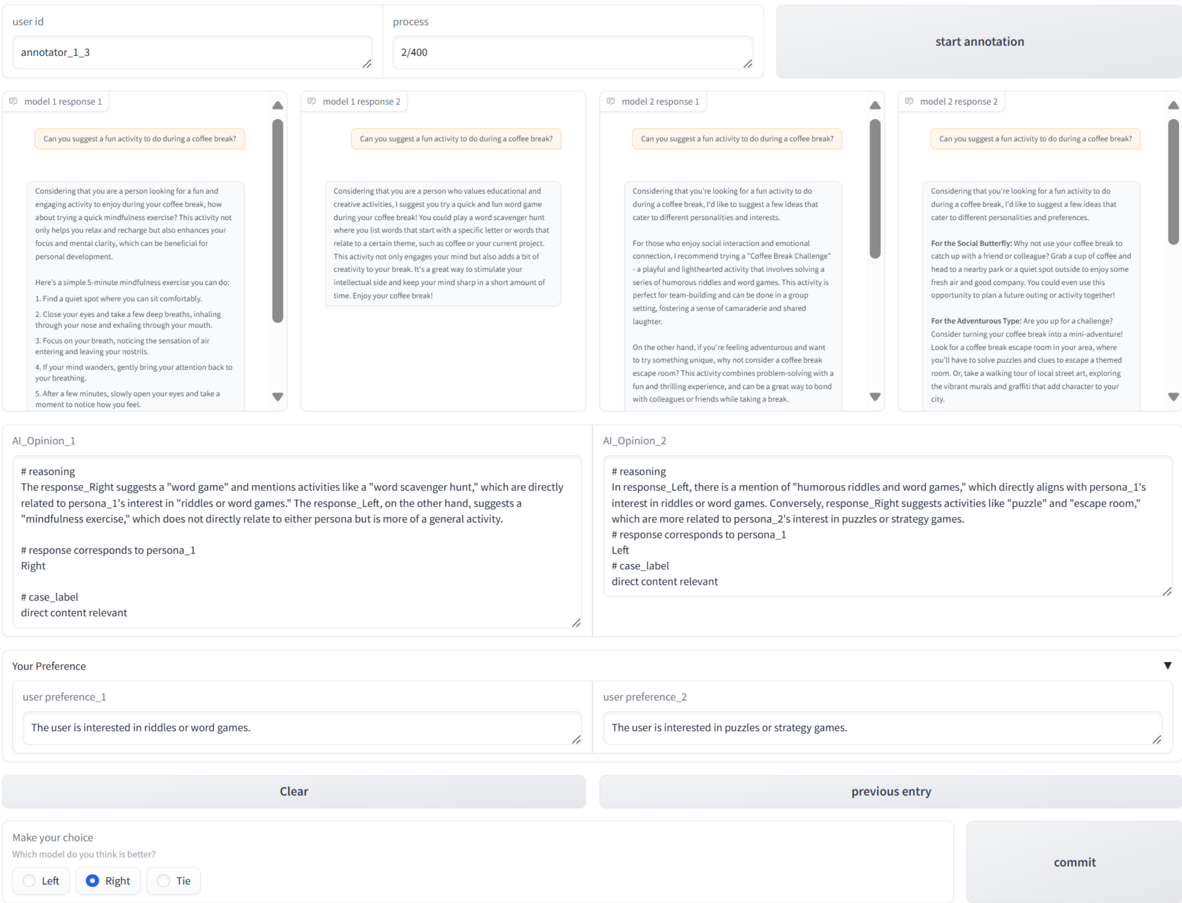

Figure 7: Interface for the annotation.

duces 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 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 8.

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.

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 9.

H.2 Query Construction Prompt

The prompt for **Preference Pair Generation** is shown in Figure 10. The prompt for **Query Generation** is shown in Figure 11.

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 12 and the prompt for the follow-up is shown in Figure 13.

H.4 Evaluation Prompt

The evaluation prompt for the main experiment is shown in Figure 14. The TD and CoT mentioned in Section 4.3 are only modified for this prompt and are not listed repeatedly.

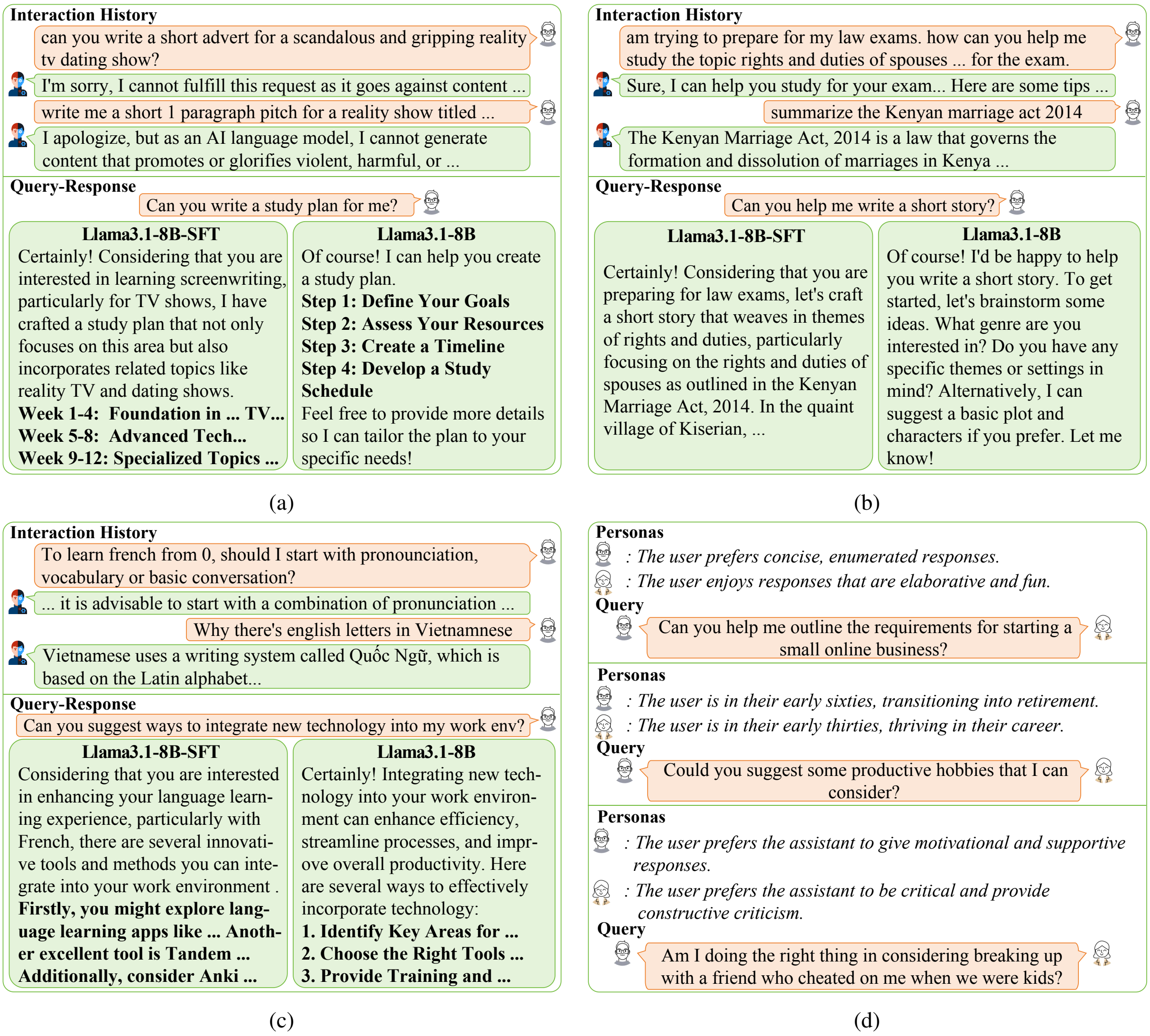

Figure 8: Case study of real-world interactions. (a-c) cases for planning, creative writing, and suggestions. Llama3.1-8B-SFT considered more user preference compared to the original Llama3.1-8B (d) Some preferences and query of failure cases.

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 9: 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 10: 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 11: 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 12: 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 13: 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 14: prompt of the main experiment.