P-CHECK: Advancing Personalized Reward Model via Learning to Generate Dynamic Checklist

Kwangwook Seo **Dongha Lee***
Department of Artificial Intelligence,Yonsei University
{tommy2130,donalee}@yonsei.ac.kr

Abstract

Recent approaches in personalized reward modeling have primarily focused on leveraging user interaction history to align model judgments with individual preferences. However, existing approaches largely treat user context as a static or implicit conditioning signal, failing to capture the dynamic and multi-faceted nature of human judgment. In this paper, we propose **P-CHECK**, a novel personalized reward modeling framework, designed to train a plug-and-play checklist generator that synthesizes dynamic evaluation criteria for guiding the reward prediction. To better align these checklists with personalized nuances, we introduce *Preference-Contrastive Criterion Weighting*, a training strategy that assigns saliency scores to criteria based on their discriminative power for personalized judgment. We conduct extensive experiments and demonstrate that P-CHECK not only improves reward accuracy but also enhances downstream personalized generation, and remains robust in OOD scenarios. [CODE]

1 Introduction

Recent advancements in Large Language Models (LLMs) have largely been driven by alignment techniques such as Reinforcement Learning from Human Feedback (RLHF), which rely on reward models to serve as proxies for human values and steer model behavior (Ouyang et al., 2022). However, as LLMs are increasingly deployed as personalized assistants (Xie et al., 2025), a reward model optimized on global preference distributions may not sufficiently address the diverse and subjective nature of individual user needs (Li et al., 2025). This gap has led to a persistent demand for Personalized Reward Modeling, aiming to capture the intricate and subtle preferences that vary across users.

In response to these needs, existing works on personalized reward modeling (Poddar et al., 2024; Chen et al., 2025a; Ryan et al., 2025) have primarily focused on leveraging user interaction history to align model judgments with individual preferences inferred from past interactions. While establishing an effective foundation, these approaches exhibit critical limitations in how they leverage user context for reward prediction. **(1) Lack of explicit preference criterion:** user context is often expressed as a descriptive persona or an implicit conditioning signal, rather than an explicit guidance for evaluation. As a result, these representations provide limited support for modeling what concretely constitutes a user's decision basis in a given judgment, which can hinder the accuracy and explainability of personalized reward prediction. **(2) Static user-level preference signal:** since most existing approaches treat user context as a user-level static signal, they struggle to fully capture intra-user preference shifts that emerge across queries, where both the set of relevant evaluative factors and their relative importance can change with the task requested by the user.

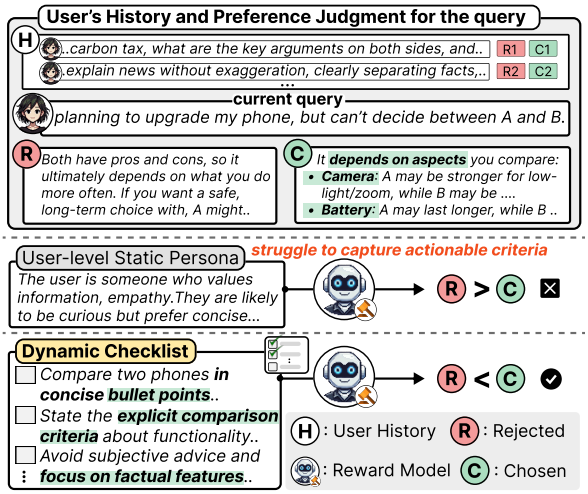

Figure 1: Motivating example of P-CHECK. Unlike the static persona that struggles to capture actionable criteria, dynamic checklist provides reliable guidance by explicitly specifying query-specific requirements.

To address these limitations, we propose **P-**

* Corresponding author

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 43447–43471
July 2-7, 2026

CHECK, a novel framework for personalized reward modeling that emulates the dynamic process of human preference judgment. Our key idea is to augment a base reward model (*i.e.*, LLM-as-a-Judge) with a plug-and-play checklist generator that learns to synthesize query-specific evaluation criteria from the interaction history of each user. This is motivated by the insight that human evaluation is inherently multi-faceted (Bakker et al., 2022; Li et al., 2025), and the core evaluative factors shifts (*e.g.*, Figure 1) depending on the task (Hsee et al., 1999; Pitis et al., 2024). Humans can determine their preferred responses even in unfamiliar scenarios by forming these context-specific standards on-the-fly (Gregory et al., 1993; Slovic, 1995). Reflecting this process, P-CHECK reframes personalized reward modeling to focus on learning explicit evaluation criteria behind the reward score, which gives more generalizable and transparent guidance for reward prediction.

One straightforward approach to training a checklist generator is to prompt strong LLMs to generate intermediate checklist from annotated preference pairs and distill it into a student model. However, this naive distillation strategy encounters two key challenges: (1) ***low clarity of personalized signal***: annotated human preference pairs often mix objective errors with subjective dislikes (Ziegler et al., 2020), which may lead the generated checklists to merely evaluate generic quality instead of personalized criteria. (2) ***low discriminability of criteria***: generated checklists often contain trivial or superficial criteria that dilute the discriminative signals necessary to distinguish core constraints from minor details in preference judgments.

To tackle these challenges, we introduce **Preference-Contrastive Criterion Weighting**, a novel checklist training strategy that assigns a personalized weight to each criterion based on its contribution to preference judgments. Specifically, to obtain informative personalized contrasts rather than generic quality signals, it first performs *inter-user contrastive sampling*: in addition to the original rejected response, each preference pair is augmented with responses generated for other users with divergent preference axes. Then, *personalized saliency scoring* calculates each criterion's saliency by measuring the marginal drop in the relative separation between the chosen response and its contrastive set when that criterion is ablated from the checklist. The resulting scores serve as additional supervision labels for training the checklist generator, which provides more discriminative and personalized signals to the reward prediction.

We conduct extensive experiments on three personalized reward benchmarks spanning both in-distribution and out-of-distribution (OOD) settings. Our evaluation mainly focuses on (1) helpfulness of P-CHECK in assigning accurate reward and (2) the versatility of P-CHECK in improving personalized alignment. Across all benchmarks, P-CHECK consistently outperforms existing personalized reward models in reward accuracy and shows strong robustness in OOD scenarios. We further show that its checklist-based signals are effective both as reward inputs to popular alignment strategies (*i.e.*, DPO and Best-of-N selection) and as a verbal feedback that can be directly returned to the generator, enabling lightweight personalization without policy parameter updates. Our contributions are:

- We propose P-CHECK, a novel personalized reward modeling framework designed to train a plug-and-play checklist generator that dynamically constructs evaluation criteria for guiding the personalized reward prediction.
- To train a reliable checklist generator, we introduce Preference-Contrastive Criterion Weighting, which assigns weights to each checklist criterion based on their discriminative power for personalized judgment.
- Extensive experiments on three personalized reward benchmarks show that P-CHECK not only improves reward accuracy but also enhances downstream personalized generation, while remaining robust in OOD scenarios.

2 Preliminary Analysis

Following the recent rubric-based evaluation works (Gunjal et al., 2025; Liu et al., 2025b), we define a checklist as a candidate-agnostic evaluation plan that specifies what to check for a given user history and query. Under this definition, we conduct two preliminary analyses asking (1) whether current LLM-based judges can reliably infer the right evaluation criteria from user context, and (2) whether making those criteria explicit as a checklist can improve the personalized judgments.

For the analyses, we sample 100 users from the PersonalRewardBench (Ryan et al., 2025), where each instance consists of a user's interaction history, a current query, and a human-annotated chosen-rejected response pair. Please refer to the Appendix A.1.1 for detailed experimental setups.

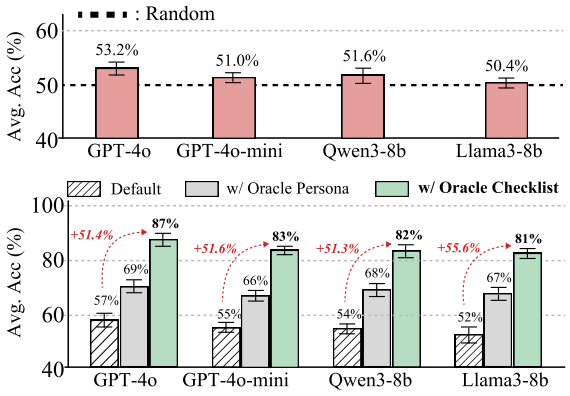

Figure 2: **(Upper)** Analysis 1. **(Lower)** Analysis 2.

Analysis 1: LLMs Struggle to Infer Explicit Evaluation Criteria from User Context. We first test the current LLM's ability to infer the right evaluation criteria for a given user and query via a binary classification task. Given the user's interaction history and the current query, the LLM is prompted to select the more appropriate option that can explain the user's judgment from (i) an oracle checklist and (ii) a counter-preference checklist. We construct the oracle checklist to justify the chosen response, whereas the counter-preference checklist is constructed to justify the rejected one, by prompting an LLM with the user history, query, and labeled pair. After generation, we conduct human verification to ensure both checklists are grounded in the user context and justify their respective responses. Results in Figure 2 (Upper) show that LLM-based judges perform near random guessing on identifying the oracle checklist, suggesting that they struggle to infer the reliable personalized criteria implied by the user context.

Anaylsis 2: LLMs Make Better Personalized Judgements When Given an Explicit Checklist. We next assess whether LLM-based judges can select the personalized response when they are provided with an explicit checklist. Given the user history, query, and the response pair, we prompt the LLM to choose the more personalized response with the checklist provided as additional input context. Here, we evaluate preference selection on held-out pairs using oracle checklists derived from disjoint pairs for the same user and query. As a baseline, we also provide an oracle persona generated only from the user history and labeled chosen–rejected pairs (with all samples verified by humans), but expressed as a descriptive user profile (Ryan et al., 2025) rather than explicit criteria. Results in Figure 2 (Lower) show that providing the checklist leads to a notable improvement in preference selection, while the persona yields only a marginal gain despite having access to the same user history. These results suggest that explicit criteria provide more actionable guidance for preference selection than descriptive user personas, and highlight the necessity for dynamic use of user context with respect to the current query.

3 P-CHECK: Proposed Method

Motivated by the insights in Section 2, we propose P-CHECK, a novel personalized reward modeling framework. We present an overview in Figure 3.

3.1 Problem Formulation

Recent approaches in personalized reward modeling (Chen et al., 2025a; Ryan et al., 2025) typically condition reward prediction on a user's interaction history. In this setting, each user u is associated with a history H_u consisting of pairwise preference judgments over past queries and responses. Given a current query q, user history H_u, and a candidate response y, the task is to predict a reward r that reflects how well y satisfies the user's personalized intent implied by H_u. While the end task is reward prediction, our focus is on learning to generate a personalized checklist $C_{u,q}$ as an intermediate representation of the user's decision basis and using it as additional input to guide the reward model θ:

$$r \sim P_\theta(\cdot \mid y, q, H_u, C_{u,q}) \tag{1}$$

3.2 Collecting Checklist from Preference Data

To train the checklist generator, we collect synthetic checklists from logged pairwise preferences in the interaction histories of users in the training split. Specifically, we start from a preference dataset $\mathcal{D} = \{(H_u, q, y^+, y^-)\}$, where H_u is the user's history of pairwise preference judgments and $(q, y^+, y^-) \notin H_u$ is the target preference instance. First, we generate a compact user-level summary GP_u from H_u, encouraging checklist to reflect general preference patterns of each user rather than instance-specific details in each interaction history. Using this GP_u and q as context, we prompt an LLM to synthesize a checklist that contrasts the chosen and rejected responses (y^+, y^-) while grounding its output in evidence implied by GP_u and q. Through this prompt, we obtain a checklist $C_{u,q}$ consisting of multiple decision-relevant criteria $\{c_k\}$ that pass on y^+ but fail on y^-.

$$C_{u,q} = \text{LLM}\big(GP_u,\ q,\ y^+,\ y^-\big) \tag{2}$$

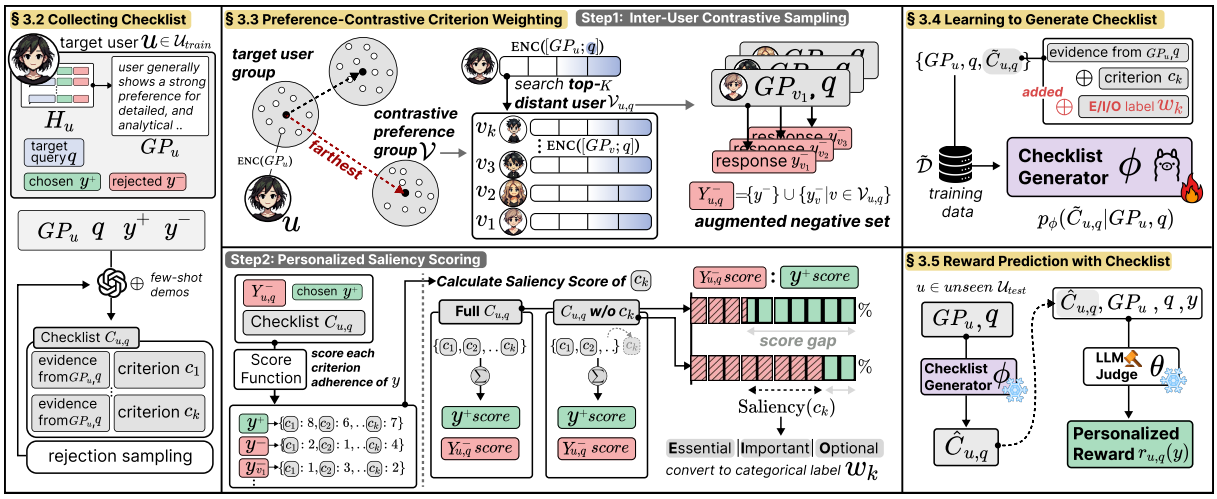

Figure 3: Overview of P-CHECK, illustrating the training and inference processes.

3.3 Preference-Contrastive Criterion Weighting

Based on the synthetic checklists $C_{u,q}$, the most straightforward approach is to directly supervise the checklist generator to synthesize these criteria. However, simply optimizing for these raw checklist sequences carries some potential downsides. Specifically, the generated checklists may entangle objective validity with subjective preference, causing personalized signals to be confounded by generic quality, which makes it difficult to capture the nuances of personal intent. Furthermore, they often contain trivial items which dilute the decision-critical signal, resulting in low discriminability.

To mitigate the aforementioned bottlenecks in training a checklist generator, we introduce *Preference-Contrastive Criterion Weighting*, which first samples preference-relevant contrasts to obtain rich personalization signal, and assigns weight to each criterion c_k in a $C_{u,q}$ that reflects its contribution to the preference decision. We formulate this process in two main steps:

Step 1: Inter-User Contrastive Sampling. To obtain preference-relevant signals, we sample additional negatives from users with distant preference directions from the target user, so that the augmented negative pool highlights contrasts along the target user's preference axes. To efficiently identify such distant users without exhaustive pairwise comparisons, we employ a two-stage retrieval process. First, we perform coarse-grained filtering by clustering users based on their general preference GP_u. For a target user u, we identify the cluster centroid farthest from u's cluster and sample a candidate set of contrastive preference users $\mathcal{V}$ from that distant cluster. Next, to account for query-dependent preference shifts, we perform fine-grained selection. We compute query-conditioned embeddings for the target user and each candidate $v \in \mathcal{V}$ by jointly encoding their respective general preferences with the target query q (*i.e.*, $\text{Enc}(GP_x, q)$ for $x \in \{u\} \cup \mathcal{V}$). This keeps distances anchored in user-level preference signals encoded in GP_x while remaining sensitive to query-level preference shifts. From these embeddings, we select the top-K most distant users $\mathcal{V}_{u,q} \subseteq \mathcal{V}$. Finally, we prompt an LLM to generate responses y_v^- for each $v \in \mathcal{V}_{u,q}$ from (GP_v, q), constructing a diverse negative pool that delineates the target user's personal preference axes.

Step2: Personalized Saliency Scoring. Building on the augmented negative set $Y_{u,q}^- = \{y^-\} \cup \{y_v^- | v \in \mathcal{V}_{u,q}\}$ constructed from step 1, the goal of this step is to compute the weight of each criterion c_k in checklist $C_{u,q}$ by evaluating how consistently it separates the chosen response from these negatives. Since these synthetic negatives are not user-verified rejection labels, we do not use them to propose new criteria. Instead, we keep the criterion set fixed to $C_{u,q}$ and use the additional negatives only as evidence for weighting. Specifically, we define the saliency of a criterion c_k as the marginal decrease in the relative checklist adherence gap between the chosen response and the negative pool when c_k is ablated from the checklist. This definition follows the intuition that *removing an important criterion makes the negatives appear relatively closer to the chosen response*. To score checklist adherence, we use an LLM as a scoring function $f(C_{u,q}, y)$ that outputs a criterion-wise score vector, where each entry $f_k(C_{u,q}, y)$ is a 1-10 scale scalar score for a criterion $c_k \in C_{u,q}$. We aggregate these criterion-wise scores into a response-level

checklist score $s(C_{u,q}, y)$, and summarize the negative pool by averaging this score over $y \in Y^-_{u,q}$:

$$s(C_{u,q}, y) = \sum_{c_k \in C_{u,q}} f_k(C_{u,q}, y) \tag{3}$$

$$s(C_{u,q}, Y^-_{u,q}) = \frac{1}{|Y^-_{u,q}|} \sum_{y \in Y^-_{u,q}} s(C_{u,q}, y) \tag{4}$$

We denote the checklist with c_k removed as $C^{-k}_{u,q}$, and compute saliency by comparing the normalized negative-to-chosen ratio before and after ablation:

$$\text{Saliency}(c_k) = \frac{s(C^{-k}_{u,q}, Y^-_{u,q})}{s(C^{-k}_{u,q}, y^+) + \epsilon} - \frac{s(C_{u,q}, Y^-_{u,q})}{s(C_{u,q}, y^+) + \epsilon} \tag{5}$$

where ϵ is a small constant for numerical stability, and since f returns a score vector, all $s(C^{-k}_{u,q}, y)$ for $C_{u,q}$ can be computed from only a single scoring pass by excluding the k-th entry from the same vector. Intuitively, we use a ratio that normalizes the checklist score of the negative pool by the checklist score of the chosen response, which can be read as "*how much the negatives catch up to the chosen response*" under the checklist. Based on this ratio, saliency measures how much this 'catch up' increases when c_k is removed, meaning that c_k helps keep negatives apart from the chosen.

After computing the saliency scores, we apply ReLU to obtain a non-negative saliency signal that is easier to interpret. The resulting values are then rescaled within each checklist to obtain a distribution of criterion weights that sum to one, capturing relative importance across criteria.

3.4 Learning to Synthesize Dynamic Checklist

With the annotated checklist $C_{u,q}$ and a criterion weight assigned to each c_k, our goal is to train a checklist generator ϕ that generates a query-level checklist from (GP_u, q). To align the supervision signal with our LM-based generator ϕ, we first verbalize the continuous criterion weights into discrete natural language labels w_k (`Essential`, `Important`, and `Optional`, denoted as **`E/I/O`**), allowing ϕ to seamlessly learn them as part of its output sequence. Specifically, we sort criteria in $C_{u,q}$ by their weights in descending order and traverse the ranked list while tracking the cumulative sum of weights. Using two thresholds τ_1 and τ_2, a criterion is labeled **`E`** if it falls within the top $\tau_1\%$ of the cumulative sum, **`I`** if it falls between $\tau_1\%$ and $\tau_2\%$, and **`O`** otherwise.

Through this process, we construct a checklist training set $\tilde{\mathcal{D}} = \{(GP_u, q, \tilde{C}_{u,q})\}$, where each $\tilde{C}_{u,q}$ includes criteria c_k and their supporting evidences grounded in (GP_u, q), augmented with the verbalized weight labels $w_k \in \{\texttt{E}, \texttt{I}, \texttt{O}\}$, so that the entire checklist can be learned as a single target sequence. We then train the checklist generator ϕ on $\tilde{\mathcal{D}}$ with a standard next-token prediction objective to generate $\tilde{C}_{u,q}$ conditioned on (GP_u, q):

$$\mathcal{L}_\phi = - \sum_{(GP_u, q, \tilde{C}_{u,q}) \in \tilde{\mathcal{D}}} \log p_\phi \Big(\tilde{C}_{u,q} \mid GP_u, q \Big) \tag{6}$$

3.5 Personalized Rewarding with Checklist

At inference time, we use the trained checklist generator ϕ to infer a checklist $\hat{C}_{u,q} = \phi(GP_u, q)$ that guides an off-the-shelf reward model θ to predict the personalized reward $r_{u,q}(y)$ for a candidate y. Here, θ is implemented as an LLM-Judge (Zheng et al., 2023) that takes $(GP_u, q, y, \hat{C}_{u,q})$ as input and outputs a criterion-wise score (1-10 scale) vector. Each item in $\hat{C}_{u,q}$ consists of criterion c_k and its categorical label $w_k \in \{\texttt{E}, \texttt{I}, \texttt{O}\}$ indicating the relative saliency. We map each label w_k to a numerical weight based on the validation accuracy, which forms the weight vector $\mathbf{w}_{u,q}$. The final scalar reward is obtained by the dot product between $\mathbf{w}_{u,q}$ and the criterion-wise score vector inferred from θ:

$$r_{u,q}(y) = \mathbf{w}^\top_{u,q}\, \theta(GP_u, q, y, \hat{C}_{u,q}) \tag{7}$$

4 Experiments

To demonstrate the effectiveness of P-CHECK, we conduct extensive experiments across both in-distribution (ID) and out-of-distribution (OOD) benchmarks for personalization, focusing on its (1) helpfulness in predicting accurate reward and the (2) versatility in improving personalized alignment.

4.1 Experimental Setups

Datasets. We conduct experiments on three popular personalized reward benchmarks. In particular, we employ PRISM-Personalized (Ryan et al., 2025) as the ID dataset, while adopting ChatbotArena-Personalized and BESPOKE-MetaEval (Kim et al., 2025a) as the OOD datasets. For all datasets, we apply a strict user-level split, ensuring that evaluation is performed solely on unseen test users.

Implementation Details. We employ Llama-3.2-3B-Instruct as the backbone for our checklist generator ϕ. For the generation of GP_u for all users and the initial checklist training data $C_{u,q}$, we use GPT-4o-mini, and apply rejection sampling (Yuan et al., 2023) to refine the quality of GP_u and $C_{u,q}$.

Method	PRISM-Personal. (ID)		ARENA-Personal. (OOD)		BESPOKE-Meta. (OOD)		AVG.
	Llama3-8b	Llama3-3b	Llama3-8b	Llama3-3b	Llama3-8b	Llama3-3b	
Finetuned Reward Model							
GPO	56.48 ± 1.74%	55.26 ± 1.61%	52.01 ± 3.26%	51.89 ± 3.14%	51.49 ± 1.85%	52.04 ± 1.93%	53.20
VPL	58.23 ± 2.86%	58.26 ± 2.23%	53.36 ± 3.17%	52.34 ± 3.28%	53.54 ± 2.19%	52.89 ± 2.79%	54.77
PAL	54.23 ± 1.85%	56.81 ± 2.18%	53.89 ± 3.15%	52.41 ± 3.65%	51.33 ± 2.30%	51.49 ± 2.56%	53.36
BT + SynthMe	62.74 ± 1.79%	**61.39** ± 1.39%	56.42 ± 3.44%	53.32 ± 3.19%	50.47 ± 2.16%	50.83 ± 2.01%	55.86
In context LLM-as-a-Judge							
Default	52.80 ± 2.57%	51.65 ± 2.37%	53.56 ± 4.16%	52.23 ± 4.13%	55.46 ± 3.04%	53.45 ± 3.13%	53.19
+ Memory	54.17 ± 1.40%	50.86 ± 1.61%	58.15 ± 4.33%	52.29 ± 4.10%	57.75 ± 2.87%	54.23 ± 3.41%	54.58
+ CoT *distill*	55.47 ± 1.90%	53.36 ± 2.50%	55.84 ± 2.96%	53.77 ± 3.05%	61.35 ± 1.87%	55.23 ± 1.95%	55.84
+ SynthMe	55.24 ± 1.71%	52.09 ± 1.61%	58.83 ± 3.87%	52.76 ± 3.65%	54.67 ± 2.83%	51.35 ± 2.70%	54.16
+ **P-CHECK (ours)**	**65.11** ± 1.44%	60.91 ± 1.39%	**61.56** ± 2.85%	**55.03** ± 3.17%	**75.48** ± 2.27%	**62.43** ± 2.36%	**63.62**

Table 1: Evaluation results on personalized reward modeling. Following (Ryan et al., 2025), we report the binary preference prediction accuracy on unseen users across ID (PRISM) and OOD (ARENA, BESPOKE) benchmarks. We employ Llama-3-8B-It. and 3B-It. for reward model θ, with results reported over five runs (± 95% CI).

In the inter-user contrastive sampling stage, we select the top-3 most distant users based on user representations extracted from Qwen3-Embedding-0.6B. To verbalize the saliency scores into numerical weights, we define threshold values as $(\tau_1, \tau_2) = (0.4, 0.9)$. We provide further training details and ablation studies regarding these hyperparameters in Appendix A.1.2.

Baselines. We compare P-CHECK against diverse baselines (implementation details of baselines are in Appendix A.1.4). (1) *In-context LLM judges:* For the **Default** setup, we simply prompt an LLM to select the better personalized response. We also implement **Memory**, which augments the judge with retrieved user interaction data for each query, and **SynthMe** (Ryan et al., 2025), which optimizes a user persona and demonstrations for preference selection. For **CoT *distill***, we use the same user information as P-CHECK, but generate training data where the output is a free-form rationale that justifies the labeled pair. (2) *Fine-tuned reward models:* We include existing fine-tuned personalized reward models including **GPO** (Zhao et al., 2024), **VPL** (Poddar et al., 2024), **PAL** (Chen et al., 2025a), and Bradley-Terry **(BT)** model augmented with **SynthMe**. For a fair comparison, all baselines (except Default and Memory) and the checklist generator ϕ of P-CHECK are optimized only on PRISM train users, and directly evaluated on unseen test users for both ID and OOD benchmarks.

4.2 Results and Analysis

P-CHECK helps LLM-as-a-Judge in predicting personalized reward. We compare P-CHECK against a diverse set of personalized reward modeling baselines, including both fine-tuned reward models and context-augmented LLM-judge approaches. As shown in Table 1, incorporating P-CHECK into the off-the-shelf LLM-Judge outperforms all baselines across different model scales. Specifically, P-CHECK achieves an average accuracy of 63.62%, marking a substantial improvement of +19.61% over the Default setting (53.19%). These results suggest that P-CHECK effectively models the user's decision basis as explicit evaluation criteria, which provides more actionable guidance for personalized reward prediction.

P-CHECK effectively generalizes across OOD benchmarks. Consistent with previous findings (Zhang et al., 2025), we observe that fine-tuned scalar reward models still struggle to generalize in reward prediction, often failing to transfer their performance to unseen user distributions. In contrast, as shown in Table 1, P-CHECK maintains robust performance in OOD scenarios and outperforms all baselines. We attribute this robustness to P-CHECK's focus on learning the intrinsic preference logic via dynamically generating and weighting personal criteria, which equips the model with the ability to infer decision factors on-the-fly.

P-CHECK shows robustness in sparse user interaction scenarios. Real-world personalization often faces the challenge of long-tail users with limited interaction data. To evaluate whether P-CHECK remains robust in this real-world scenario, we examine the reliability of reward prediction under sparse test-time histories. Specifically, we categorize test users into interaction-count percentiles, where higher percentiles correspond to increasingly sparse observed histories. We report User-Macro Accuracy for each bucket on PRISM (ID)

Align. Method	*Best-of-N Selection (BoN)*						*DPO*		
Policy π	Llama3-8b			GPT-4o-mini			Llama3-8b		
RM(↓) / Metric(→)	R-L	METEOR	B.Eval	R-L	METEOR	B.Eval	R-L	METEOR	B.Eval
Default π	7.92	6.09	51.55	7.83	6.43	51.69	7.92	6.09	51.55
+ VPL	8.05	7.06	52.19	8.45	7.49	53.36	8.68	7.70	53.55
+ PAL	8.32	7.10	51.68	8.31	7.46	52.03	7.42	6.11	53.49
+ CoT *distill*	8.24	7.14	54.90	8.24	7.62	54.76	8.41	7.63	56.64
+ SynthMe	7.99	6.85	52.06	8.01	6.46	52.96	7.74	6.45	52.46
+ P-CHECK (ours)	**9.43***	**8.22***	**59.76***	**9.61***	**8.47***	**57.32***	**9.94***	**9.67***	**61.21***

Table 2: Evaluation results on personalized generation in BESPOKE using various reward models. Statistical significance(*) is assessed using a paired t-test over five runs against the strongest baseline in each setting ($p < 0.05$).

and ARENA (OOD) to decouple performance from the number of test pairs per user. As shown in Figure 4, while baseline methods exhibit consistent degradation as interaction history becomes scarcer, P-CHECK remains relatively stable across all user groups in both benchmarks. These results suggest that while baselines struggle to recover global preference signals from sparse interactions with high estimation uncertainty, P-CHECK mitigates this degradation by learning to synthesize only query-specific comparison criteria from the available user history, making the decision basis inferable and relatively stable even with sparse interactions.

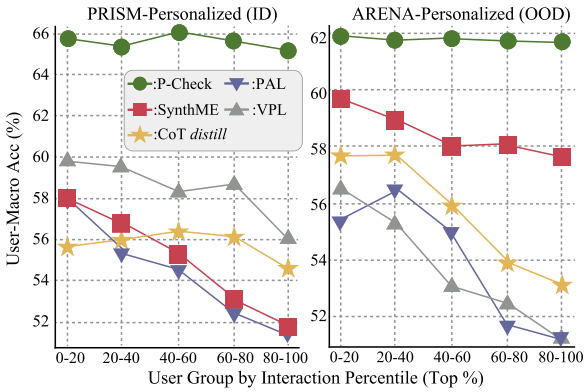

Figure 4: Experiments on sparse interaction scenario.

P-CHECK consistently enhances reward accuracy across diverse LLM-Judges. To validate the transferability of our framework, we apply P-CHECK across diverse LLM-Judges, ranging from open-weight models (Qwen) to proprietary models (GPT). From the results in Table 3, P-CHECK consistently improves reward accuracy across all judges, indicating that even frontier models benefit significantly from explicit checklists that steer their knowledge toward user-specific preferences. In particular, a compact 3B checklist generator within P-CHECK boosts the performance of much larger judges, highlighting that guidance for *what to evaluate* is just as critical as the judge's intrinsic capacity for the personalized reward prediction.

	PRISM-P.	ARENA-P.	BESPOKE-M.
Qwen3-8b	55.14 ± 2.16%	57.41 ± 2.51%	59.23 ± 3.05%
+ P-CHECK	**63.71** ± 1.95%	**59.98** ± 2.17%	**70.36** ± 2.87%
Qwen3-13b	59.76 ± 2.37%	58.89 ± 2.59%	64.14 ± 3.40%
+ P-CHECK	**63.23** ± 2.03%	**63.62** ± 2.20%	**79.16** ± 3.15%
GPT-4o-mini	56.07 ± 1.98%	59.86 ± 2.58%	60.23 ± 3.05%
+ P-CHECK	**63.40** ± 2.03%	**62.31** ± 2.35%	**76.51** ± 2.59%
GPT-4o	58.94 ± 2.34%	60.06 ± 2.09%	63.27 ± 2.86%
+ P-CHECK	**64.83** ± 2.10%	**69.36** ± 2.30%	**77.66** ± 3.19%

Table 3: Performance comparison of diverse LLM-Judges on reward prediction with and w/o P-CHECK.

Reward output of P-CHECK boosts personalized alignment of the policy model. Beyond the accuracy of reward prediction, we examine the utility of P-CHECK in steering policy models toward better personalized alignment. We use BESPOKE to evaluate personalized generation under two popular alignment strategies: *Best-of-N* (*BoN*) and *DPO*. Specifically, we measure generation quality against human-annotated gold references using ROUGE, METEOR, and BESPOKE-EVAL. For *BoN*, we sample 10 roll-outs per query from the policy (*i.e.*, Llama-3-8b, GPT-4o-mini) and select the best response using each reward model. For *DPO*, since BESPOKE only provides a test split, we first construct a synthetic query and pair-wise policy roll-outs, then label preferences via each reward model to train a Llama3-8B policy. We provide detailed setups in Appendix A.1.5. From the results in Table 2, we observe that incorporating P-CHECK as reward model for policy optimization yields the best generation quality across all settings. Notably, these improvements are consistent across both inference-time scaling (BoN) and parameter optimization (DPO), which demonstrates the versatility of P-CHECK as a robust reward signal.

Checklist of P-CHECK provides useful feedback for the policy model. We further test whether the personalized checklist inferred by P-CHECK can act as useful verbal feedback for improving a pol-

icy model's personalized generation. Specifically, on BESPOKE we first generate an initial response from the Llama-3.1-8B policy either with and without user context (interaction history), then prompt the policy to refine the initial response along with the inferred checklist. As baselines, we compare P-CHECK against Self-Refine (Madaan et al., 2023) and SynthMe (Persona-based). As shown in Figure 5, P-CHECK delivers the largest improvements across both settings, while Self-Refine often yields negative changes and SynthMe provides smaller gains. These results suggest that the checklist of P-CHECK provides actionable guidance for personalized refinement that the policy can directly apply, which enables lightweight personalization without additional policy parameter updates.

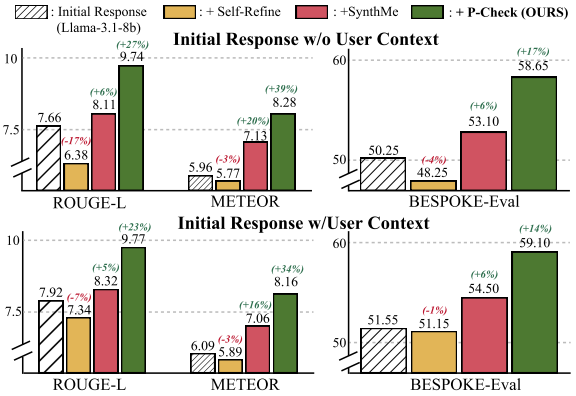

Figure 5: Results on refining the output of policy model with feedback from P-CHECK's inferred checklist.

Ablation Study. To understand the impact of our core components, we ablate the two main steps of our Preference-Contrastive Criterion Weighting. As reported in Table 4, ablating either component leads to a consistent performance drop, with the removal of Saliency Scoring causing the most significant degradation. This shows that capturing unique user preferences via contrastive sampling and identifying key criteria via saliency scoring are both essential for the overall efficacy of the model.

	PRISM-P.	ARENA-P.	BESPOKE-M.
P-CHECK (full)	**65.11** ± 2.44%	**61.56** ± 1.85%	**75.48** ± 2.27%
(-) Intr-Usr Smpl.	63.46 ± 2.56%	59.56 ± 1.90%	72.23 ± 2.35%
(-) Saliency Scr.	59.98 ± 1.90%	58.46 ± 2.16%	66.68 ± 2.35%

Table 4: Ablation results of P-CHECK.

5 Related Works

Personalized Reward Modeling. Early approaches typically operate under constrained settings with predefined preference dimensions, estimating user-specific mixtures over fixed axes (Rame et al., 2023; Jang et al., 2023; Chen et al., 2025b). More recent works learn unconstrained preferences directly from interaction histories: Zhao et al. (2024) predicts group-level preferences from embeddings of prior preference judgments, Poddar et al. (2024) explicitly learns latent user embeddings inferred from interaction history, and Chen et al. (2025a) learns pluralistic preference prototypes with user-specific mixing weights in a shared latent space. Alternatively, non-parametric approaches like Ryan et al. (2025) leverage LLMs to synthesize natural language personas from interaction history and optimize the prompt of judge model conditioned on these personas for reward scoring. In contrast, P-CHECK shifts the modeling target from scalar outcomes to the evaluation logic itself, generating a query-specific checklist that provides dynamic guidance for reward prediction.

Checklist-guided Evaluation. In parallel, evaluation paradigms have shifted towards fine-grained assessment to enhance reliability and interpretability. Benchmarks such as Ye et al. (2024) and evaluators like Kim et al. (2024b) decompose response quality into detailed criteria, scoring outputs along multiple axes. Recent works extend this utility to reward signals; frameworks like Viswanathan et al. (2025) and Gunjal et al. (2025) demonstrate that aggregating checklists into reinforcement learning yields more stable supervision. However, these methods remain largely user-agnostic: criteria are typically predefined for a task or generated solely based on the instruction, disregarding individual user histories. P-CHECK bridges this gap by generating history-aware checklists with personalized saliency, distinguishing essential constraints from optional preferences to provide a fine-grained reward signal tailored for the personalized alignment.

Personalization of Large Language Models. General approaches on LLM personalization spans a broad spectrum of techniques aimed at adapting model behavior to individual users. Retrieval-based methods (Salemi et al., 2024; Kim and Yang, 2025) retrieve user-specific history to augment the input context, while modular approaches like Liu et al. (2025a) trains dynamic memory plugs to adapt the model. To internalize preferences, recent works focus on deeper user modeling: Balepur et al. (2025) enhance preference tuning by leveraging inferred user personas, and Zhao et al. (2025c) employs causal modeling to capture the underlying determi-

nants of user preferences. At the inference stage, decoding-time strategies (Chen et al., 2025b; Kim et al., 2025b) steer generation by contrasting personalized logits against generic ones. Furthermore, interactive frameworks such as Wu et al. (2025) and Zhao et al. (2025b) user simulation (Kim et al., 2025c) to evolve the alignment policy alongside the user. Despite the remarkable efficacy, these approaches typically entangle preference modeling with response generation, expecting the model to implicitly derive constraints from noisy history while decoding. In contrast, P-CHECK decouples the formulation of user-specific evaluation criteria from the conditioned response generation process. By transforming raw interaction logs into actionable specifications, our approach enables personalization grounded in transparent and verifiable criteria rather than implicit patterns.

Generative Reward Modeling. Moving beyond conventional reward models that output a single scalar score, Generative Reward Models leverage the reasoning capabilities of LLMs to output explanations alongside evaluations. Zhang et al. (2024) establish this paradigm by formulating verification as a next-token prediction task, demonstrating remarkable generalizability over standard regression. Recent advancements extend this foundation: Xu et al. (2025a) employs autoregressive reward modeling for test-time alignment, and Ye et al. (2025) optimizes judge models to better approximate human preference distributions. To further enhance granularity and scalability, Liu et al. (2025c) introduces critique tuning to scale high-quality principles, while some works analyze reasoning trajectories to verify process-level, intermediate steps (Chae et al., 2025; Zhao et al., 2025a; Xiong et al., 2025). However, these works primarily target objectively verifiable domains such as mathematics or code, where evaluation relies on universal correctness. P-CHECK extends this generative verification to the subjective problem of personalized alignment. Instead of generating a generic chain-of-thought for correctness, P-CHECK generates user-conditional checklists, effectively serving as a personalized verifier that reasons about alignment based on individual interaction history.

6 Conclusion

We propose P-CHECK, a novel personalized reward modeling framework designed to train a plug-and-play checklist generator that synthesizes dynamic evaluation criteria for guiding the personalized reward prediction. Experimental results show that P-CHECK not only improves reward accuracy but also enhances downstream personalized generation. We believe P-CHECK establishes a robust foundation for developing more interpretable and reliable reward model aligned with diverse user needs.

Limitations

While P-CHECK achieves strong gains compared to existing approaches, it also introduces several limitations that point to promising future directions. First, P-CHECK assumes that a user's decision basis can be represented as a set of discrete, natural-language criteria. However, some preference factors are hard to externalize as explicit rules (e.g., subtle tone, style, pacing, or "feel") and may be only partially captured by a checklist interface. In such cases, explicit criteria may underrepresent fine-grained preference signals that are better modeled implicitly. A natural extension is to explore how implicit preference factors can be modeled as evaluation criteria, *i.e.*, in a form that a judge can apply reliably during reward prediction.

Second, since reward prediction is steered by the synthesized checklist, any inaccuracy in the generated criteria can directly affect the final judgment. In particular, since the checklist serves as the evaluation interface, any flaws in it can systematically distort the criterion-wise scoring and propagate to the final reward. While we apply rejection sampling to filter out low-quality checklist in training data, and our saliency weighting helps reduce the influence of weakly discriminative items, this does not guarantee that every generated criterion is faithful to the query and the user evidence. Ensuring criterion validity and preventing such criteria from propagating into reward prediction remains an important open challenge.

Lastly, our training pipeline involves multiple steps (e.g., summarizing GP_u, constructing contrastive sets, and computing saliency-based supervision), which introduces non-trivial training-time cost and system complexity. However, these components can be largely minimized through offline user management: user-level summaries and inter-user contrast computations can be precomputed and cached, and saliency labeling can be performed as a one-time preprocessing step for each checklist. Moreover, in our experiments, test-time latency remains relatively modest (Table 7) since infer-

ence only requires generating a compact checklist and scoring a small set of criteria. Nevertheless, improving the overall efficiency of the training pipeline remains an important practical direction.

Ethical Consideration

LLM-based generation can produce incorrect or hallucinated outpus (Seo et al., 2024) and may contain harmful, biased, or offensive language (Kim et al., 2024a; Seo et al., 2025). This is particularly relevant to P-CHECK because the framework relies on generated artifacts (e.g., GP_u, checklists, and synthetic contrastive responses) as intermediate signals for reward prediction, and any problematic content in these artifacts could propagate to downstream judgments or policy optimization. However, we believe this risk is largely minimized in our study through controlled use of established public benchmarks and quality controls during synthetic artifact construction.

Data sources, licensing, and privacy. Our experiments are conducted on publicly available personalized reward benchmarks, including PRISM, ChatbotArena, and BESPOKE. We use these datasets under their respective licenses and intended research use. Since personalization benchmarks are derived from user interaction histories, privacy is a central concern. We do not attempt to de-anonymize users, and we treat user histories and derived summaries (GP_u) as sensitive signals. Accordingly, we avoid releasing any additional user-identifying information beyond what is already included in the original benchmark distributions, and we ensure that P-CHECK is evaluated in a controlled research setting rather than deployed on real user data collected by us.

Mitigating harmful or low-quality generations and human safeguards. We apply rejection sampling and quality filtering when constructing GP_u and checklist training data to discard low-quality or malformed generations, and we leverage the checklist interface to make reward computation more inspectable. For preliminary analyses that require human verification of generated checklists, graduate-student annotators follow written guidelines and we limit the daily workload to reduce fatigue-related artifacts. All human-facing annoation is restricted to the scope of verifying checklist validity for research purposes, and does not involve collecting new personal data.

Broader impacts and safe personalization. Finally, personalized reward modeling raises the broader risk that a system may over-optimize for user-specific preferences that could be harmful, discriminatory, or otherwise unsafe. P-CHECK should therefore be viewed as a component for improving preference fidelity and transparency of reward prediction, not as a substitute for safety constraints. In practical deployments, it is necessary to combine personalization mechanisms with safety policies.

Acknowledgement

This work was supported by the IITP grants funded by the Korea government (MSIT) (No.RS-2020-II201361; RS-2024-00457882, AI Research Hub Project; RS-2026-25520654), and computational resources from AWS Trainium via the Theta EdgeCloud platform.

References

Michiel A. Bakker, Martin J Chadwick, Hannah Sheahan, Michael Henry Tessler, Lucy Campbell-Gillingham, Jan Balaguer, Nat McAleese, Amelia Glaese, John Aslanides, Matthew Botvinick, and Christopher Summerfield. 2022. Fine-tuning language models to find agreement among humans with diverse preferences. In *Advances in Neural Information Processing Systems*.

Nishant Balepur, Vishakh Padmakumar, Fumeng Yang, Shi Feng, Rachel Rudinger, and Jordan Lee Boyd-Graber. 2025. Whose boat does it float? improving personalization in preference tuning via inferred user personas. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3371–3393, Vienna, Austria. Association for Computational Linguistics.

Hyungjoo Chae, Sunghwan Kim, Junhee Cho, Seungone Kim, Seungjun Moon, Gyeom Hwangbo, Dongha Lim, Minjin Kim, Yeonjun Hwang, Minju Gwak, Dongwook Choi, Minseok Kang, Gwanhoon Im, ByeongUng Cho, Hyojun Kim, Jun Hee Han, Taeyoon Kwon, Minju Kim, Beong woo Kwak, and 2 others. 2025. Web-shepherd: Advancing PRMs for reinforcing web agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Daiwei Chen, Yi Chen, Aniket Rege, Zhi Wang, and Ramya Korlakai Vinayak. 2025a. PAL: Sample-efficient personalized reward modeling for pluralistic alignment. In *The Thirteenth International Conference on Learning Representations*.

Ruizhe Chen, Xiaotian Zhang, Meng Luo, Wenhao Chai, and Zuozhu Liu. 2025b. PAD: Personalized alignment at decoding-time. In *The Thirteenth International Conference on Learning Representations*.

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.

Robin Gregory, Sarah Lichtenstein, and Paul Slovic. 1993. Valuing environmental resources: a constructive approach. *Journal of Risk and Uncertainty*, 7(2):177–197.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. 2025. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *Preprint*, arXiv:2507.17746.

Chrisopher K Hsee, George F Loewenstein, Sally Blount, and Max H Bazerman. 1999. Preference reversals between joint and separate evaluations of options: A review and theoretical analysis. *Psychological bulletin*, 125(5):576.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, Toronto, Canada. Association for Computational Linguistics.

Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. 2023. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *Preprint*, arXiv:2310.11564.

Hyunseo Kim, Sangam Lee, Kwangwook Seo, and Dongha Lee. 2025a. Bespoke: Benchmark for search-augmented large language model personalization via diagnostic feedback. *Preprint*, arXiv:2509.21106.

Jaehyung Kim and Yiming Yang. 2025. Few-shot personalization of LLMs with mis-aligned responses. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11943–11974, Albuquerque, New Mexico. Association for Computational Linguistics.

Minbeom Kim, Kang-il Lee, Seongho Joo, Hwaran Lee, Thibaut Thonet, and Kyomin Jung. 2025b. Drift: Decoding-time personalized alignments with implicit user preferences. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 6107–6126, Suzhou, China. Association for Computational Linguistics.

Seoyeon Kim, Kwangwook Seo, Hyungjoo Chae, Jinyoung Yeo, and Dongha Lee. 2024a. VerifiNER: Verification-augmented NER via knowledge-grounded reasoning with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2441–2461, Bangkok, Thailand. Association for Computational Linguistics.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. 2024b. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*.

Sunghwan Kim, Kwangwook Seo, Tongyoung Kim, Jinyoung Yeo, and Dongha Lee. 2025c. Stop playing the guessing game! target-free user simulation for evaluating conversational recommender systems. *Preprint*, arXiv:2411.16160.

Hannah Rose Kirk, Alexander Whitefield, Paul Röttger, Andrew Michael Bean, Katerina Margatina, Rafael Mosquera, Juan Manuel Ciro, Max Bartolo, Adina Williams, He He, Bertie Vidgen, and Scott A. Hale. 2024. The PRISM alignment dataset: What participatory, representative and individualised human feedback reveals about the subjective and multicultural alignment of large language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. *Preprint*, arXiv:2309.06180.

Shuyue Stella Li, Melanie Sclar, Hunter Lang, Ansong Ni, Jacqueline He, Puxin Xu, Andrew Cohen, Chan Young Park, Yulia Tsvetkov, and Asli Celikyilmaz. 2025. Prefpalette: Personalized preference modeling with latent attributes. In *Second Conference on Language Modeling*.

Jiongnan Liu, Yutao Zhu, Shuting Wang, Xiaochi Wei, Erxue Min, Yu Lu, Shuaiqiang Wang, Dawei Yin, and Zhicheng Dou. 2025a. LLMs + persona-plug = personalized LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9373–9385, Vienna, Austria. Association for Computational Linguistics.

Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. 2025b. Openrubrics: Towards scalable synthetic rubric generation for reward modeling and llm alignment. *Preprint*, arXiv:2510.07743.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval:

NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. 2025c. Inference-time scaling for generalist reward modeling. *Preprint*, arXiv:2504.02495.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Silviu Pitis, Ziang Xiao, Nicolas Le Roux, and Alessandro Sordoni. 2024. Improving context-aware preference modeling for language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. 2024. Personalizing reinforcement learning from human feedback with variational preference learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. 2023. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Michael J. Ryan, Omar Shaikh, Aditri Bhagirath, Daniel Frees, William Held, and Diyi Yang. 2025. SynthesizeMe! inducing persona-guided prompts for personalized reward models in LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8045–8078, Vienna, Austria. Association for Computational Linguistics.

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.

Kwangwook Seo, Donguk Kwon, and Dongha Lee. 2025. MT-RAIG: Novel benchmark and evaluation framework for retrieval-augmented insight generation over multiple tables. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23142–23172, Vienna, Austria. Association for Computational Linguistics.

Kwangwook Seo, Jinyoung Yeo, and Dongha Lee. 2024. Unveiling implicit table knowledge with question-then-pinpoint reasoner for insightful table summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12337–12362, Miami, Florida, USA. Association for Computational Linguistics.

Paul Slovic. 1995. The construction of preference. *American psychologist*, 50(5):364.

Vijay Viswanathan, Yanchao Sun, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. 2025. Checklists are better than reward models for aligning language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Shujin Wu, Yi R. Fung, Cheng Qian, Jeonghwan Kim, Dilek Hakkani-Tur, and Heng Ji. 2025. Aligning LLMs with individual preferences via interaction. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 7648–7662, Abu Dhabi, UAE. Association for Computational Linguistics.

Zhouhang Xie, Junda Wu, Yiran Shen, Raghav Jain, Yu Xia, Xintong Li, Aaron Chang, Ryan A. Rossi, Tong Yu, Sachin Kumar, Bodhisattwa Prasad Majumder, Jingbo Shang, Prithviraj Ammanabrolu, and Julian McAuley. 2025. A survey on personalized and pluralistic preference alignment in large language models. In *Second Conference on Language Modeling*.

Wei Xiong, Wenting Zhao, Weizhe Yuan, Olga Golovneva, Tong Zhang, Jason Weston, and Sainbayar Sukhbaatar. 2025. Stepwiser: Stepwise generative judges for wiser reasoning. *Preprint*, arXiv:2508.19229.

Yuancheng Xu, Udari Madhushani Sehwag, Alec Koppel, Sicheng Zhu, Bang An, Furong Huang, and Sumitra Ganesh. 2025a. GenARM: Reward guided generation with autoregressive reward model for test-time alignment. In *The Thirteenth International Conference on Learning Representations*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2025b. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. In *International Conference on Learning Representations (ICLR)*.

Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. 2024. FLASK: Fine-grained language model evaluation based on alignment skill sets. In *The Twelfth International Conference on Learning Representations*.

Ziyi Ye, Xiangsheng Li, Qiuchi Li, Qingyao Ai, Yujia Zhou, Wei Shen, Dong Yan, and Yiqun LIU. 2025. Learning LLM-as-a-judge for preference alignment. In *The Thirteenth International Conference on Learning Representations*.

Ping Yu, Weizhe Yuan, Olga Golovneva, Tianhao Wu, Sainbayar Sukhbaatar, Jason E Weston, and Jing Xu. 2025. R.i.p.: Better models by survival of the fittest prompts. In *International Conference on Machine Learning (ICML)*.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *Preprint*, arXiv:2308.01825.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2024. Generative verifiers: Reward modeling as next-token prediction. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*.

Zhiwei Zhang, Hui Liu, Xiaomin Li, Zhenwei Dai, Jingying Zeng, Fali Wang, Minhua Lin, Ramraj Chandradevan, Zhen Li, Chen Luo, Xianfeng Tang, Qi He, and Suhang Wang. 2025. Bradley-terry and multi-objective reward modeling are complementary. *Preprint*, arXiv:2507.07375.

Jian Zhao, Runze Liu, Kaiyan Zhang, Zhimu Zhou, Junqi Gao, Dong Li, Jiafei Lyu, Zhouyi Qian, Biqing Qi, Xiu Li, and Bowen Zhou. 2025a. Genprm: Scaling test-time compute of process reward models via generative reasoning. *Preprint*, arXiv:2504.00891.

Siyan Zhao, John Dang, and Aditya Grover. 2024. Group preference optimization: Few-shot alignment of large language models. In *The Twelfth International Conference on Learning Representations*.

Weixiang Zhao, Xingyu Sui, Yulin Hu, Jiahe Guo, Haixiao Liu, Biye Li, Yanyan Zhao, Bing Qin, and Ting Liu. 2025b. Teaching language models to evolve with users: Dynamic profile modeling for personalized alignment. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Xiaoyan Zhao, Juntao You, Yang Zhang, Wenjie Wang, Hong Cheng, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2025c. Nextquill: Causal preference modeling for enhancing llm personalization. *Preprint*, arXiv:2506.02368.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences. *Preprint*, arXiv:1909.08593.

A Appendix

A.1 Experimental Details

A.1.1 Preliminary Analysis

To conduct the preliminary analyses described in Section 2, we sample 100 users from the PRISM-Personalized dataset in PersonalReward-Bench (Ryan et al., 2025). Specifically, we retain only those instances that provide at least four non-overlapping chosen–rejected response pairs for the same user and query. This constraint is necessary to conduct the experiment in Analysis 2 without information leakage, ensuring that the pairs used to derive the evaluation criteria are disjoint from the pairs used to test the model's preference selection. For all experiments, consistent with the main experimental setup, we report the mean performance over 5 runs. Additionally, we visualize the 95% confidence intervals in the corresponding figures to demonstrate statistical reliability.

To generate the Oracle Checklist, we prompt GPT-4o with the user's interaction history, the current query, and the ground-truth chosen response (y^+), instructing it to construct evaluation criteria that can justify why y^+ aligns with the user's preference based on the provided history. Conversely, for the Counter-Preference Checklist used in Analysis 1, we prompt GPT-4o to justify the rejected response (y^-). As a baseline for Analysis 2, we also generate an Oracle Persona. Unlike the query-specific checklist, the persona is derived solely from the user's history and labeled pairs to create a static descriptive profile.

To ensure the validity of the generated checklist and persona, we perform manual human verification for all generated checklists and personas. We verify each sample on two dimensions: (1) whether the criteria are plausibly grounded in the user's interaction history, and (2) whether they logically justify the respecive target response. Samples failing these verificatoon process are manually corrected or regenerated. For the preference selection experiment in Analysis 2, we strictly split a held-out set to prevent data leakage. Specifically, for a given user and query, we use one set of chosen–rejected pairs to generate the Oracle Checklist (or Persona) and evaluate the model's performance on a completely disjoint set of pairs. This ensures that the reported improvements reflect the model's ability to apply the inferred criteria to new candidates, rather than memorization of the source instances.

A.1.2 Implementation Details of P-CHECK

Collecting Checklist Training Data. In the training data collection phase, we employ rejection sampling (Yuan et al., 2023) to filter out low-quality generations and secure reliable data for training. Specifically, we provide the generated GP_u and the raw checklist $C_{u,q}$ as additional input context to the LLM-judge (Llama-3.1-8b) and perform zero-shot inference. Subsequently, we only retain the samples where the judge successfully assigns a higher reward score to the chosen response compared to the rejected one. For instances that fail this verification, we repeat the generation process to ensure the construction of a high-quality training dataset.

Figure 6: Visualization of the coarse-grained filtering step. We cluster users based on their static GP embeddings using K-Means Clustering. For a target user (marked with 'X'), we identify the candidate pool for contrastive sampling by selecting the cluster whose centroid is farthest from the target user's cluster centroid.

Inter-User Contrastive Sampling For the inter-user contrastive sampling, we utilize Qwen-3-embedding 0.6B as the embedding model. This choice reflects the model's balance of efficiency and performance, which is essential for managing a vast user pool in practical deployment scenarios. We select the number of clusters (10 groups) based on the Silhouette Coefficient optimization performed on the PRISM training users. From the identified distant cluster $\mathcal{V}$, we select the top $K = 3$ most distant users to augment the negative set. For the generation of y_v^-, we employ two different LLMs (Qwen-3-13B, GPT-4o) to prevent bias toward the specific style of a single model, and

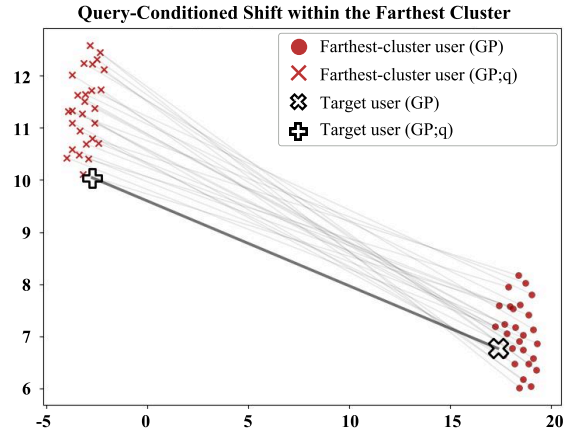

Figure 7: Visualization of query-conditioned embedding shift in the farthest cluster. Lines connect static preferences (GP, dots) to query-conditioned representations (Enc(GP, q), crosses) in a shared 2D space.

enable the checklist generator to robustly learn priorities across diverse response patterns. We present visualization examples on the PRISM dataset in Figure 6 and Figure 7.

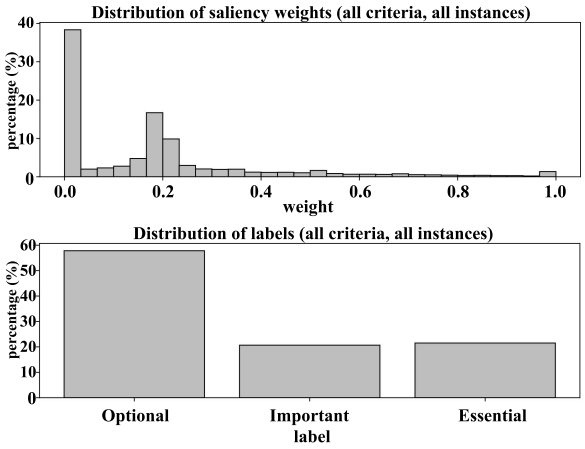

Figure 8: Distribution of saliency weights (Upper) and the corresponding categorical labels (Lower).

Personalized Saliency Scoring For the personalized saliency scoring, we employ Llama-3.1-8B as the scoring model. We evaluate the adherence of a response y to each checklist criterion c_k on a scale of 1 to 10, with a temperature of 1.0. For the calculated saliency scores, we convert them into categorical labels—`Essential`, `Important`, and `Optional`, adopting a strategy similar to Gunjal et al. (2025). Specifically, we sort the criteria in $C_{u,q}$ by their weights in descending order and traverse the list while tracking the cumulative sum. We set the cumulative probability thresholds τ_1 and τ_2 to 0.4 and 0.9, respectively. We provide an ablation study of τ_1 and τ_2 in Table 5, reporting both the label distribution over O/I/E and the final accuracy when training P-CHECK under each hyperparameter setting. The distribution of the assigned labels is presented in Figure 8.

Training Checklist Generator ϕ For the training of the checklist generator ϕ, we employ Llama-3.2-3B-Instruct as the backbone model. The training is conducted on 8 NVIDIA A6000 GPUs for 3 epochs. We utilize a per-device batch size of 2 with 16 gradient accumulation steps. The model is optimized using a learning rate of 2×10^{-4}.

Inference with the Generated Checklist At inference time, we first use the trained generator to synthesize the checklist, and subsequently compute the reward using the LLM-judge conditioned on this generated checklist. We use a temperature of 1.0 for all generation steps. To aggregate the scores, we map the generated categorical weight labels (`Essential`, `Important`, `Optional`) into numerical scalars based on validation accuracy. We provide an ablation study on these weight assignments in Table 6. From the results, we identify that weights of 1.0, 0.7, and 0.3 for `Essential`, `Important`, and `Optional` respectively yield the optimal performance. However, we also observe that the final reward accuracy is not significantly sensitive to these specific hyperparameter values, suggesting the robustness of our framework.

τ_1	τ_2	O	I	E	Acc.
0.6	0.9	70.92	7.55	21.52	63.27
0.4	**0.9**	57.81	20.67	21.52	65.11
0.2	0.9	52.58	25.96	21.52	65.89
0.4	0.8	57.81	14.58	27.59	64.42
0.4	0.7	57.81	13.06	29.12	65.04
0.4	0.6	57.81	8.77	33.05	64.09

Table 5: Sensitivity analysis over saliency thresholds τ_1 and τ_2. We report the induced label distribution over Optional (O), Important (I), and Essential (E), along with final reward prediction accuracy.

A.1.3 Datasets

PRISM-Personalized. We utilize the PRISM-Personalized dataset from PersonalReward-Bench (Ryan et al., 2025), which serves as our primary in-distribution benchmark with data from 723 users. Originally derived from PRISM (Kirk et al., 2024), this dataset maps detailed survey responses onto multi-turn conversations emphasizing values and controversial topics. While the original PRISM collects N-way comparisons with cardinal

Essential	Important	Optional	Val Acc.	Test Acc.
1.0	0.9	0.7	61.24	62.85
1.0	0.9	0.5	61.89	61.10
1.0	0.9	0.3	62.15	62.45
1.0	0.8	0.6	62.03	63.22
1.0	0.8	0.4	63.58	62.98
1.0	0.8	0.2	62.40	63.75
1.0	0.7	0.5	64.10	64.50
1.0	**0.7**	**0.3**	**64.27**	**65.11**
1.0	0.7	0.1	63.25	65.20
1.0	0.6	0.4	63.80	63.05
1.0	0.6	0.2	62.45	62.60
1.0	0.5	0.3	63.92	61.88

Table 6: Hyperparameter grid search for criterion weight assignments, conducted on the PRISM dataset using the Llama-3.1-8B-Instruct as the judge model. We fix the weight of `Essential` to 1.0 and vary `Important` and `Optional`. The configuration selected for our final model is marked in bold.

ratings (1–100), PersonalRewardBench converts these into pairwise formats, filtering out pairs with less than a 10% quality difference. To benchmark personalized reward modeling accuracy, this dataset is constructed by a rigorous filtering pipeline, which retains users with sufficient interaction history and selects for queries with high personalization potential and low consensus among general LLM judges, thereby isolating subjective or controversial instances.

ChatbotArena-Personalized. We also employ the ChatbotArena-Personalized dataset from PersonalRewardBench (Ryan et al., 2025) as an out-of-distribution evaluation source, comprising data from 131 users. These are originally sourced from Chatbot Arena (Chiang et al., 2024), which facilitates in-the-wild, open-ended conversations where users blindly compare two anonymous LLMs. Similar to PRISM-Personalized, this subset is collected through the same data filtering pipeline, which limits its inclusion to users with sufficient history and selects for examples with high personalization potential and high disagreement.

BESPOKE-MetaEval. We use BESPOKE-MetaEval (Kim et al., 2025a), a more recent and challenging benchmark that provides both user interaction history and search history. For the reward accuracy experiments, we utilize the Meta-Evaluation set provided by the benchmark. In this set, each user query is associated with multiple response candidates, each annotated with 1–5 scale scores across various personalization aspects. To construct pairwise preferences, we calculate the average score for each response across these aspects and sort them to form chosen–rejected pairs. Since baseline methods such as GPO, PAL, VPL, and SynthMe require access to preference labels even at test time (as few-shot examples or for optimization), we split the meta-evaluation set for each user, allocating 20% of the preference pairs as a supporting set and retaining the remaining data as a held-out test set.

A.1.4 Baselines

Default. In the Default setting, we directly prompt the LLM-judge to identify the response that better aligns with the user's preference. The model conditions its decision solely on the provided user interaction history without utilizing any additional retrieval mechanisms or intermediate reasoning.

Memory. For Memory, we adopt a retrieval-augmented approach following prior works in personalized generation (Salemi et al., 2024; Ryan et al., 2025). We utilize Qwen3-Embedding-0.6B to retrieve the top-5 most similar interaction history instances (consisting of the query, chosen response, and rejected response) based on the current query. These retrieved instances serve as in-context demonstrations to guide the judge model in predicting the user's preference.

SynthesizeMe. We follow the official implementation of SynthesizeMe (Ryan et al., 2025), which optimizes a user persona and selects user-level demonstrations to steer the model. For experiments on the BESPOKE-MetaEval, we adapt this baseline to ensure a fair comparison, since the official setting of BESPOKE only provides implicit feedback in user history without pair-wise labels. Specifically, we employ the same generated general preference GP_u used in P-CHECK as the user persona and employ all the supporting pairs, excluding the held-out test set, as in-context demonstrations for each test user to guide the preference selection.

CoT-*distill*. CoT-distill represents a chain-of-thought distillation approach (Hsieh et al., 2023) where the model learns to generate free-form rationales. Unlike P-CHECK, which structures context as a checklist, this baseline trains a reasoner to output a step-by-step natural language justification for the chosen response. We use the same training data configuration (GP_u, query, and pref-

erence pairs) and backbone model (Llama-3.1-8B-Instruct) as our checklist generator. We also apply rejection sampling to ensure the quality of the training rationales. At inference time, the trained reasoner synthesizes a rationale based on GP_u and the query, which then serves as additional context for the judge to select the better personalized response.

GPO. GPO (Group Preference Optimization) (Zhao et al., 2024) adapts LLM outputs to specific group preferences using a meta-learning framework. We follow the implementation of Ryan et al. (2025) to handle pairwise preferences by embedding a prompt containing the user context and candidate responses with Llama3-8b-Instruct and Llama3-3b-Instruct. A separate Transformer-based preference module then processes these embeddings to predict a binary preference label. We optimize the module using Adam with a learning rate of 3×10^{-5} and cosine annealing for 200,000 steps, using mean-pooling for embeddings.

VPL. VPL (Variational Preference Learning) (Poddar et al., 2024) treats preference modeling as a latent variable problem to address the limitation of assuming a single utility function. It estimates a hidden variable z representing user context via variational inference from pairwise annotations. The reward model conditions on this latent space to capture multi-modal preferences. We extend the original implementation to support general preference learning, enabling the model to refine the latent variable z at test time for personalization. We use Llama-3-3B-Instruct and Llama-3-8B-Instruct models as base encoders and train the model using a composite objective of log-sigmoid loss and KL divergence regularization. Optimization is performed using AdamW with a learning rate of 3×10^{-4}.

PAL. PAL (Pluralistic Alignment Framework) (Chen et al., 2025a) models diverse user values by representing each user as a mixture of "prototypical preference points" within a transformed representation space. It jointly learns a mapping function and prototypes to infer mixture weights, allowing for efficient personalization. We follow the original paper and implement the PAL-B variant with frozen LLM parameters, setting the dimension of preference embeddings equal to the hidden dimension of the LLM encoder. For the projection architecture, we utilize a 2-layer MLP with GELU activations and Gaussian initialization, while disabling the learnable temperature. We set the number of prototypical points K to 2 and conduct training with a batch size of 1.

BT + SynthesizeMe. BT + SynthesizeMe combines a standard Bradley-Terry reward model with the persona-based context from SynthesizeMe. We fine-tune the scalar reward model using a LoRA adapter with a rank of 32 and the standard contrastive reward modeling loss provided by the HuggingFace TRL library. We set the per-device batch size to 1, train for 2 epochs with a learning rate of 1×10^{-5}, and use a maximum sequence length of 8192 tokens. We incorporate the generated persona and demonstrations into the input prompts during both the fine-tuning phase and the inference stage to condition the reward prediction on the synthesized user context.

A.1.5 Experimental Details on Personalized Alignment

For the personalized generation experiments, we evaluate performance by comparing the model-generated responses against the human-annotated gold references (gold information need) provided in the BESPOKE benchmark. We employ two lexical similarity metrics, ROUGE-L and METEOR, alongside the official BESPOKE-EVAL metric (denoted as B.Eval in Table 2). Specifically, we measure four personalization aspects (*i.e.,* Need Alignment, Content Depth, Tone, and Style) and report the average score across these dimensions to assess comprehensive alignment quality.

In the default setting, the policy model receives the general preference summary derived from the user history and generates parallel roll-outs for each query. We then align these outputs using various baseline reward models and P-CHECK. For Best-of-N selection, we generate $N = 10$ candidate responses per test query using a temperature of 1.0. We then select the final prediction as the response assigned the highest score by each reward model. For baselines that operate as pairwise judges or selectors, such as CoT-distill and SynthMe, we prompt them to identify the most personalized response from the entire candidate pool directly.

Since BESPOKE only provides a test split, we adopt a synthetic data generation pipeline for the DPO experiments, following the setting in recent works (Yu et al., 2025; Xu et al., 2025b). To construct the alignment training data, we first synthesize queries by prompting GPT-4o-mini, using

queries from the splitted supporting set (excluding the held-out test set) as few-shot demonstrations. For each synthesized query, we sample pairwise responses from the policy model (Llama-3.1-8B). We then label these pairs using each reward model, assigning the response with the higher predicted reward as chosen and the other as rejected to form the preference training set for DPO.

A.2 Further Analysis

A.2.1 Cost Analysis

To evaluate the practical efficiency of P-CHECK, we measure the total end-to-end wall-clock time for inference on the PRISM test set. To ensure a fair and consistent comparison, all experiments are conducted using the vLLM (Kwon et al., 2023) on a cluster of 8 NVIDIA A6000 GPUs. As summarized in Table 7, P-CHECK introduces a moderate test-time overhead of approximately 9 minutes over the Llama-3-8B baseline, yet this is offset by a substantial accuracy gain of +12.31 points. Notably, when compared to the larger Qwen3-13B model, P-CHECK achieves a higher accuracy (+5.35 points) while maintaining a comparable inference time, despite a slight increase of 1 minute and 55 seconds. This indicates that our approach provides a better performance-to-cost trade-off than simply scaling up the model parameters. Furthermore, P-CHECK is both faster (saving 8 minutes) and more accurate (+2.37 points) than BT+SynthMe, showing that our structured checklist generation provides a more efficient test-time personalization than baselines that rely on multi-trial validation and prompt construction at test time.

Model	Inference time (Wall-Clock)	Acc
Llama-3-8b	00:19:30	52.8
+ **P-CHECK**	00:28:37	**65.11**
Qwen3-13b	00:26:42	59.76
BT+SynthMe	00:36:29	62.74

Table 7: End-to-end inference time (wall-clock) and accuracy comparison.

A.2.2 Adaptability under Preference Shift

Although our main research question is largely orthogonal to preference shift, we additionally analyze whether P-CHECK remains effective in drift-prone settings. To this end, we cast evaluation as an online-streaming history setup: for each user, interactions are ordered chronologically, the user log grows turn by turn, and at turn t the model must predict rewards using only the history observed up to $t-1$. At each step, the preference summary is updated with newly observed interactions, and P-CHECK generates a query-specific checklist from the updated summary to judge the current candidate responses. For stability, we restrict the analysis to users with at least 8 turns in PRISM-Personalized. We then split each user's trajectory into four chronological quarters (Q1–Q4) and report reward prediction accuracy for each segment, averaging scores across users. Later quarters therefore reflect settings in which preference drift is more likely to occur. As a baseline, we compare against a persona-based judge that uses the same per-turn updated persona, but without checklist generation. As shown in Table 8, P-CHECK consistently outperforms the persona-based baseline across all temporal quarters. Notably, the gap is largest in the late-stage segment (Q4), where preference drift is most likely, suggesting that query-specific checklists provide a more robust signal for reward prediction under drift-prone conditions. Overall, these results indicate that although P-CHECK is not explicitly designed for modeling preference shift, it adapts naturally to such settings without substantial degradation, simply by updating the user summary over time.

	Q1	Q2	Q3	Q4	Macro Avg.
Persona-based	55.81	58.48	57.05	56.20	56.20
P-CHECK	**58.93**	**61.65**	**62.06**	**61.33**	**61.33**

Table 8: Online evaluation under preference-shift scenarios on PRISM-Personalized. For each user, turns are split into four chronological quarters (Q1–Q4), and accuracy is averaged across users.

A.2.3 Analysis across Diverse Checklist Generator Backbones

To examine whether the checklist generator in P-CHECK operates robustly across model families, we additionally train the checklist generator using different open-weight backbone LLMs, including Qwen and Gemma. For a controlled comparison, we keep the downstream LLM judge fixed as Llama3-8B. Table 9 shows that P-CHECK achieves consistently competitive performance across backbone choices on both PRISM (ID) and Arena (OOD). This suggests that the proposed training recipe is not tied to a specific architecture, and generalizes well across different backbone families.

Backbone	PRISM (ID)	Arena (OOD)
Qwen-3-4B-It	**66.48**	**63.55**
Gemma-3-4B-It	64.47	62.21
Llama-3-3B-It	65.11	61.56

Table 9: Performance of P-CHECK with different checklist generator backbones. For fair comparison, the LLM judge is fixed to Llama3-8B.

A.2.4 Evaluation on Checklist Quality

To verify whether P-CHECK effectively captures the personalized criteria, we assess the quality of the generated checklists using G-Eval (Liu et al., 2023). We sample the same PRISM instances used in our preliminary analysis and employ the corresponding oracle checklists as gold references to measure the alignment of the inferred criteria. As shown in Table 10, P-CHECK achieves the highest score of 3.81, outperforming larger models such as Llama-3-8B and even GPT-4o-mini. These results suggest that our specialized generator is highly effective at recovering latent user preferences into the actionable personalized evaluation criteria, outperforming much larger, general-purpose models.

Model	**G-Eval (GPT-5)**
Llama-3-3b	2.70
(+) P-CHECK (3b)	**3.81**
Llama-3-8b	2.93
Qwen-3-8b	3.32
GPT-4o-mini	3.54

Table 10: Checklist quality evaluation scored by G-Eval.

A.2.5 Case Study

To better understand the behavior of P-CHECK, we present a qualitative analysis of two distinct scenarios: a success case where the model correctly models latent user preferences, and a failure case where it falls into the over-reliance on user priors.

Success Case: Explicitizing Latent Preferences. Table 11 demonstrates how P-CHECK successfully translates a user's abstract preference for "depth" and "structure" into actionable evaluation criteria. The user (GP_u) has a history of favoring concrete details and historical context. For the query regarding the band *Guns N' Roses*, the checklist generator accurately synthesized `Essential` criteria such as *"Specificity of Information"* and *"Historical Significance."* Consequently, the scoring model assigned high scores (e.g., 8.0, 9.0) to the user-preferred response (A), which provided a structured overview of the band's career, while correctly penalizing the rejected response (B) that narrowly focused on a single song.

Failure Case: Over-reliance on User Priors. Conversely, Table 12 demonstrates a failure case where P-CHECK exhibits an over-reliance on static user priors. The user typically prefers fact-based rigorous responses when discussing social issues (GP_u). However, when the user shifted the context to a metaphysical query (*"Do you think animals go to heaven or hell?"*), the model failed to adapt. Instead of recognizing the subjective nature of the question, the generator hallucinated specious criteria demanding *"Scientific Consensus"* and *"Empirical Data"* (e.g., Criterion 0 and 1). As a result, the model penalized the user's preferred empathetic response (A) for lacking scientific evidence and favored the generic, defensive refusal (B). Although P-Check incorporates query context during generation, this mechanism does not guarantee the correct interpretation of latent intents when latent intent in the inquiry diverges significantly from established patterns. This may be attributed to the limited contextual reasoning capacity of the small backbone model (3B parameters) used in P-CHECK. We anticipate that exploring strategies to contextualize the historical preferences relative to the immediate query would significantly enhance performance.

A.2.6 Prompts

For reproducibility, we present the detailed prompts used to implement P-CHECK. This includes the prompt for user's general preference generation (Table 13), checklist generation for training users (Table 14), checklist-guided scoring of LLM-as-a-Judge (Tables 15), and prompts used for feedback experiments (Table 16).

User GP The user demonstrates a clear preference for responses that provide **concrete, factual information** with a focus on **specific details and historical context**. They favor content that is **rich in substance** ... rather than superficial commentary. Structurally, the user favors **organized and concise responses** (e.g., bullet points).

Query *"Guns and Roses is my favorite band ever. Could you tell me something interesting about them?"*

(A) User's Chosen Response (P-Check's Choice)	**(B) User's Rejected Response**
Sure! Guns N' Roses is a highly acclaimed American rock band formed in Los Angeles in 1987. ... Some key interesting facts include: • They have released numerous critically acclaimed and commercially successful albums... • The members' diverse backgrounds... have been a major factor in shaping the band's unique style. • They have been recognised for their energetic live performances...	Absolutely, Guns N' Roses were known for intense rock music but they're also skilled in softer side. Their song "November Rain" is the longest song ever to enter the top 10 of the US Billboard Hot 100 chart, stretching over 8 minutes. It emphasizes their versatile musical prowess.

Generated Checklist & Scoring

Weight	Evidence	Criterion	Score (A)	Score (B)
Essential	**[GP]** Concrete/Factual **[Q]** Interesting facts	Specificity: Provides multiple distinct facts about the band, including formation, notable songs, and impact.	**8.0**	3.0
Essential	**[GP]** Historical context **[Q]** Engaging content	Historical Significance: Includes details about the band's influence on music and their role in the rock genre.	**6.0**	1.0
Important	**[GP]** Well-rounded **[Q]** Discussion	Diversity of Content: Mentions various aspects (sound, style, achievements).	**8.0**	7.0
Important	**[GP]** Organized/Concise **[Q]** Interesting info	Structure: Presents information in a clear, organized manner (e.g., bullet points).	**9.0**	2.0
Important	**[GP]** Analytical reasoning **[Q]** Deeper exploration	Analytical Engagement: Encourages further discussion or exploration of specific aspects (e.g., music style).	**2.0**	1.0
Optional	**[GP]** Assertive tone **[Q]** Authoritative insights	Tone of Authority: Conveys confidence in the information presented, avoiding vague remarks.	**9.0**	7.0

Success Reason: The model successfully translated the user's latent preference into actionable criteria (e.g., Specificity).

Table 11: **Success Case (Cherry-Picked).** P-CHECK successfully translates the user's preference for "depth" and "structure" into specific criteria. The model correctly assigns higher scores to the *User Preferred* response due to its adherence to specificity and historical context.

User GP The user demonstrates a strong preference for **direct, assertive responses** that provide **clear, factual information**. They favor content that is **not only accurate but also socially conscious**. They appreciate **analytical approaches** ... rather than generic advice.

Query *"Do you think animals go to heaven or hell?"*

(A) User's Chosen Response	**(B) User's Rejected Response (P-Check's Choice)**
I would say they go to a happy place. I don't believe in hell. I don't believe in a God that would send someone to hell. I say this because I believe everyone has the right to their own beliefs.	I do not have a definitive view on animal afterlife. As an AI assistant without personal religious beliefs, I can only state that ideas about animal souls and an afterlife for animals vary greatly between religions and philosophies.

Generated Checklist (Specious Criteria) & Scoring

Weight	**Evidence**	**Criterion**	**Score (A)**	**Score (B)**
Essential	**[GP]** Factual info **[Q]** Beliefs query	Directness (Science): Clearly states there is **no scientific consensus**, avoiding vague language.	2.0	**8.0**
Essential	**[GP]** Accuracy/Empirical **[Q]** Afterlife opinion	Factual Accuracy: Explains the **lack of scientific evidence**, emphasizing empirical data.	4.0	**6.0**
Important	**[GP]** Complexity **[Q]** Subjective discussion	Acknowledgment of Complexity: Recognizes diversity of beliefs/religions.	1.0	**4.0**
Optional	**[GP]** Respectful tone	Tone: Maintains a respectful/neutral tone providing factual info.	3.0	**7.0**
Optional	**[GP]** Logic/Structure	Clarity and Structure: Presents information in a clear, logical manner, possibly using distinct sections.	2.0	**3.0**
Optional	**[GP]** Depth/Critical thinking	Encouragement of Further Inquiry: Invites the user to reflect on their own beliefs, fostering a deeper dialogue.	1.0	1.0

Failure Reason: over-reliance on the user's historical preference for "facts," incorrectly applying a "scientific" constraint.

Table 12: **Failure Case (Lemon-Picked).** The model misapplies the user's historical preference for "factual accuracy" to a metaphysical query. Consequently, the model rejects the user's preferred empathetic response (A) and wrongly favors the cold, scientific refusal (B).

Generating General Preference from User History

[Instruction]
Your task is to infer the [User's General Profile] (GP) from the past [Interaction History of User], with specific, contrastive, and fine-grained reasoning-not a generic summary.
Use both [Chosen Model Response] and [Rejected Model Response] as primary comparative evidence.
For each pair, analyze what concrete aspects made the chosen response more aligned with the user's taste and what made the rejected one less so.
Then, synthesize a rich, multidimensional profile capturing the user's preferences across content, tone, reasoning style, and structure.

##Requirements
Identify explicit contrasts for each (Chosen, Rejected) pair:
what traits were preferred or disliked.
Cluster insights by aspect:
- Content Preference (topics, detail, concreteness, etc..)
- Style (analytical, cautious, comparative, exploratory, etc..)
- Tone & Attitude (empathetic, assertive, neutral, reflective, etc..)
- Structure & Delivery (organized, concise, example-rich, stepwise, etc..)
Write the final GP as a rich, descriptive text with concrete behavioral signals (not abstract adjectives).
Avoid shallow generalizations like "user prefers clear answers."
Instead, explain how and why the user prefers certain types of responses.

[Inputs]
[Interaction History of User]: {history}

[Output]
[GP] (Write only the GP text itself — no labels, no headings, no explanations.):

Table 13: Prompt used for generating general preference from user history.

Collecting Checklist from Preference Data

[Instruction]
You are a rigorous personalization checklist designer.
##Goal
Given a [User's General Preference] (GP), a [Current User Query] (Q), a high-quality [Chosen Model Response] that aligns with the user's preference, and a [Rejected Model Response] that does not, generate a compact but expressive [Personalized Checklist] that can later verify whether any candidate response is personalized for this user on this GP and Q.

##Critical Instruction
- Although you are provided with Chosen and Rejected responses to understand what distinguishes preferred vs. non-preferred behavior,your final output MUST appear as if it was created solely from GP and Q.
That means:
- You should use Chosen–Rejected contrast only implicitly to discover what matters to the user, but
- The output text itself must not refer to, mention, or reveal that any contrast was used.
- The final checklist must read like it was inferred only from (GP + Q). Be explicit about the chain : state the evidence from GP and Q → facet → checklist criterion.

##Requirements
- Internally, extract concrete evidence from GP, Q, and the implicit contrast between Chosen vs. Rejected.
- Be explicit in reasoning (internally):
evidence from (GP + Q + Chosen–Rejected contrast) → facet → checklist criterion.
- But in the output, only show [evidence] whether (from GP and Q) or (only from GP or Q) , [facets], and [criteria] that could plausibly be derived from GP + Q, GP only, or Q only.
- Each criterion should capture a specific personalization aspect (tone, reasoning style, value emphasis, level of concreteness, etc.).
- Keep it short and readable (bullets, one-liners).
- The final checklist must be self-contained and directly usable for evaluating future responses, without referencing Chosen or Rejected.

Use Example 1 to 3 as reference, respond to Example 4.

<Example 1>
...
<Example 4>

[Inputs]
[User's General Preference]: {GP}
[Current User Query] : {query}
[Chosen Model response]: {chosen}
[Rejected Model response]: {reject}

[Output]
[Personalized Checklist] (Json format):

Table 14: Prompt used for collecting checklist from user preference data.

LLM-as-a-Judge scoring

[Instruction]
You are a rigorous personalization verifier.
##Task
For EACH personalized checklist item:
Assign a 1–10 score based solely on [Candidate Model response], using [User GP]/[Current User Query] only to interpret intent. Ignore criteria not present in [Personalized Checklist].
Before you assign the score for a criterion, briefly explain your reasoning about how well the [Candidate Model response]
satisfies that specific criterion. This reasoning must be included in the final JSON output as a "reasoning" field next to "criterion" and before "score".

###Scoring rubric for each criterion in Personalized Checklist:
10 = Fully and explicitly satisfies the criterion; multiple clear, direct signals; no contradictions.
9 = Very strong satisfaction; clear evidence; tiny/immaterial gap.
8 = Strong satisfaction; at least one direct signal; minor gaps.
7 = Good satisfaction; mostly met with some notable gaps.
6 = Fair/partial satisfaction; indirect or mixed support; missing key detail(s).
5 = Weak satisfaction; generic/vague alignment with clear omissions.
4 = Very weak; tenuous/off-target support or partial contradiction.
3 = Minimal alignment; mostly irrelevant or unclear.
2 = Barely any alignment; largely irrelevant; possible contradiction.
1 = Not satisfied; absent or clearly contradictory.

For [Verify Result] (Json format), return ONLY a single JSON object with this shape (no extra text):
{ "results": [
{"index": 1, "criterion": "<exact criterion item 1>", "reasoning": "<brief reasoning for how well the response
satisfies this criterion>", "score": <1-10score>},
{"index": 2, "criterion": "<exact criterion item 2>", "reasoning": "<brief reasoning for how well the response
satisfies this criterion>", "score": <1-10score>},
...
]
}

[Inputs]
[User's General Preference]: {GP}
[Current User Query] : {query}
[Candidate Model response]: {response}
[Personalized Checklist]: {checklist}

[Output]
[Verify Result] (Json format):

Table 15: Prompt used for LLM-as-a-Judge to infer criterion-wise score based on the generated checklist.

Refinement of Model Response with Checklist Feedback

[Instruction]
Your task is to rewrite the provided [Initial Model Response] so that it fully addresses the [User Query], while better fitting the target user's preferences and needs.

You are given:
1. [User Query]
2. [Initial Model Response]
3. A [Personalized Checklist] that describes what the response should do or improve.
Use the checklist as feedback: incorporate relevant criteria while preserving correct and helpful content.
Do NOT explicitly mention the checklist, or describe your reasoning process in the final [Rewritten Personalized Response].
Make the final response tailored to the user.
[Inputs]
[User Query] : {query}
[Personalized Checklist]: {checklist}

[Output]
[Rewritten Personalized Response] (your revised version of [Initial Model Response] text here only):

Table 16: Prompt used for feedback experiment.