Test-Time Personalization: A Diagnostic Framework and Probabilistic Fix for Scaling Failures

Linhai Zhang
King's College London
linhai.zhang@kcl.ac.uk

Yulan He*
King's College London & The Alan Turing Institute
yulan.he@kcl.ac.uk

Abstract

Existing approaches to LLM personalization focus on constructing better personalized models or inputs, while treating inference as a single-shot process. In this work, we study **Test-Time Personalization (TTP)** along an unexplored axis: scaling inference-time computation by sampling N candidates from a personalized policy model and selecting the best with a personalized reward model. We prove that oracle selection yields expected utility growing logarithmically with the number of sampled candidates, establishing a theoretical ceiling for test-time scaling. However, standard reward models fail to realize this potential. To diagnose why, we derive a unified scaling law that decomposes any reward model's Best-of-N curve into four measurable quantities and reveals two failure modes, *user-level collapse* (near-constant prediction for some users) and *query-level reward hacking* (negative correlation with true quality for some queries). Guided by this law, we propose a probabilistic personalized reward model whose learned variance effectively mitigates both failure modes. Experiments confirm both elements of our framework: TTP delivers consistent scaling across multiple policy models and personalized text generation tasks, and our scaling law closely matches observed scaling curves across reward-model variants.

1 Introduction

Large language models excel across diverse tasks, yet they predominantly produce *one-size-fits-all* responses that ignore individual user preferences [1]. This limitation has motivated growing research interest in LLM personalization. Existing approaches fall into three categories. *Personalized Prompting* augments input context with retrieved user history [2, 3]. *Personalized Adaptation* directly fine-tunes model parameters on user data [4, 5]. *Personalized Alignment* combines multi-objective reward models with user-specific weights [6, 7]. Despite their differences, these approaches share a common paradigm: they focus on constructing better personalized models or inputs, while treating inference as a single-shot generation process.

Meanwhile, scaling test-time computation has emerged as a powerful axis for improving LLM performance, particularly in reasoning tasks [8–10]. Recent work has begun to combine test-time scaling with personalization, but only along two axes: *scaling user interactions* to learn each user's preferences online [11], and *scaling reward-model reasoning* via generative reward models [12]. A third, equally natural axis, *scaling the policy model itself*, has not been studied for personalization. This raises a natural question: ***Can we improve personalization by scaling test-time computation for a weak personalized model?***

A straightforward instantiation of test-time personalization is parallel sampling [3, 13]: a personalized policy generates N candidates per query, and a personalized reward model selects the best. We first establish the theoretical promise of this approach: when the number of candidates increases, the expected utility of the optimal sample grows logarithmically (Section 3). This provides a theoretical

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

ceiling for what test-time personalization can achieve. Realizing this ceiling in practice, however, is far from trivial. Scaling the policy is necessary but not sufficient. The gains are bottlenecked by the reward model's ability to identify the best candidate. As Figure 1 shows, both a global reward model trained on pooled data and a per-user reward model trained on individual histories perform near random selection across N, leaving the oracle gap wide open. ***What prevents standard reward models from closing this gap?***

To diagnose this gap, we develop an analytical framework that connects a reward model's correlation with the golden score to its Best-of-N scaling behavior (Section 4). The analysis surfaces two distinct failure patterns, *user-level collapse* where the reward model degenerates to near-constant predictions for some users and *query-level reward hacking* where the reward model's predictions negatively correlate with quality for some queries. We then generalize the oracle scaling law into a *general expression* that forecasts the scaling curve of any reward model from four measurable quantities. Guided by this expression, we propose a **probabilistic personalized reward model** that mitigates both failure patterns via learned variance, enabling stable test-time scaling in practice (Section 5).

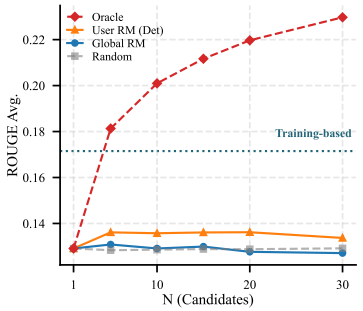

Figure 1: Test-time personalization on LaMP-4: News Headline Generation. Oracle shows logarithmic scaling that surpasses training-based baselines, while standard Reward Models (RMs) fail to scale, performing close to, or worse than, random selection.

Experiments on two benchmarks covering five personalized text generation tasks confirm both elements of the framework: our probabilistic reward model scales reliably and surpasses training-based baselines on most tasks, and the general expression closely matches observed scaling curves. In summary, our contributions are three-fold:

- We study Test-Time Personalization (TTP) along a new axis focusing on scaling policy model compute for personalized text generation.
- We detect two failure patterns and derive a predictive scaling law, a closed-form expression that forecasts any reward model's scaling curve from four measurable quantities.
- We propose a probabilistic personalized reward model that mitigates both failure patterns and scales reliably across multiple policy models and tasks.

2 Preliminaries

We focus on *personalized text generation* tasks, where each user has demonstrated stylistic preferences through written examples and the system must mimic these preferences at inference time.

2.1 Problem Formulation

Consider a set of users $\mathcal{U}$, where each user $u \in \mathcal{U}$ has an underlying preference function $r_u^* : \mathcal{Q} \times \mathcal{X} \to \mathbb{R}$ that maps a query-response pair (q, x) to a scalar reward reflecting how well x aligns with the user's preferences for query q. Each user is associated with historical data $\mathcal{D}_u = \{(q_i, x_i)\}_{i=1}^{n_u}$.

Given a query q, a personalized policy π_u conditioned on user history generates N candidate responses: $\{x_1, x_2, \ldots, x_N\} \sim \pi_u(\cdot \mid q)$. A personalized reward model $\hat{r}_u$ trained on $\mathcal{D}_u$ selects the best candidate: $x^* = \arg\max_{x_i} \hat{r}_u(q, x_i)$. The effectiveness of TTP is measured by the expected true reward of the selected response:

$$U(N) = \mathbb{E}_{q,\, x_{1:N} \sim \pi_u} \left[r_u^*(q, x^*) \right]. \tag{1}$$

The *oracle* strategy selects candidates using the true preference function r_u^*, yielding the optimal utility $U_{\text{oracle}}(N)$. The goal of TTP is to approach this oracle performance.

2.2 Experimental Setup

We use five personalized text generation tasks from LaMP [3] and LongLaMP [13], covering news headlines, scholarly titles, abstracts, product reviews, and topical posts. The personalized policy follows a retrieval-augmented generation (RAG) recipe [3], conditioning generation on user-history

examples retrieved per query. Since r_u^* is unobservable, we use ROUGE between generated responses and user-written references as a ground-truth-reward proxy. The oracle is therefore a *ROUGE-oracle*, bounding the headroom under this proxy rather than under human preference; the diagnostic framework itself is proxy-agnostic, and Section 6.4 tests transfer beyond ROUGE.

For each user, we sample candidate responses from the policy model and compute their ROUGE scores based on the golden response to construct training data. We consider two standard reward models: **Global RM**, trained on pooled data across users, and **User RM**, trained per user on $\mathcal{D}_u$. For evaluation, the policy model generates N candidates per query and different reward models select the best candidate. We report ROUGE scores averaged across users.

3 The Promise of Test-Time Personalization

We begin by establishing the theoretical foundation of TTP: given access to an oracle reward function, the expected utility of selected responses grows logarithmically with the number of candidates. We then empirically validate this scaling law and show that oracle TTP can surpass training-based methods [4].

3.1 Theoretical Foundation

Our theoretical analysis establishes a scaling law for oracle selection:

Theorem 3.1 (Oracle Scaling Law). *Assume the true reward of responses sampled from $\pi_u(\cdot \mid q)$ is sub-Gaussian with mean μ_u and variance proxy σ_u^2. Let $x^*_{\text{oracle}} = \arg\max_i r_u^*(q, x_i)$ denote the oracle selection from N i.i.d. samples. Then the expected population-level utility satisfies:*

$$\bar{U}_{\text{oracle}}(N) = \mathbb{E}_u\left[U_{\text{oracle},u}(N)\right] \leq \bar{\mu} + \bar{\sigma} \cdot c\sqrt{\ln N} \tag{2}$$

for $N \geq 2$, where $\bar{\mu} = \mathbb{E}_u[\mu_u]$, $\bar{\sigma} = \mathbb{E}_u[\sigma_u]$, and $c > 0$ is a universal constant.

The formal proof is provided in Appendix A.1. The sub-Gaussian assumption is mild and generally satisfied when the policy generates diverse but bounded-quality responses, which is typical for LLM outputs under temperature sampling; we empirically verify this property on our data in Appendix C.2. The $\sqrt{\ln N}$ form arises from the well-known result on the expected maximum of sub-Gaussian random variables, which grows as $O(\sqrt{\ln N})$ for N i.i.d. samples.

Theorem 3.1 establishes a theoretical ceiling for test-time personalization: even a weak policy (low $\bar{\mu}$) can yield high-quality outputs through scaled sampling, provided the reward model can identify the best candidates. The key question becomes whether we can build reward models that approach this oracle performance.

3.2 Empirical Validation

We validate the oracle scaling law on two representative personalization tasks: scholarly title generation (LaMP-5) and product review generation (LongLaMP). For each task, we sample $N \in \{1, 5, 10, 15, 20, 30\}$ candidates per query and select the best one using the ground-truth reward (average ROUGE-1 and ROUGE-L scores against reference).

Figure 2 presents the results. The oracle scaling curves exhibit clear logarithmic growth, closely matching the theoretical prediction of Theorem 3.1. Notably, oracle TTP surpasses training-based methods at modest candidate counts ($N \approx$ 5–10), showing that substantial gains are achievable with moderate computational overhead. These results establish that test-time personalization offers a promising pathway to performance beyond training-based methods, provided we can construct effective reward models.

4 The Challenges of Test-Time Personalization

Having established that TTP can substantially improve personalization with oracle reward models, we now investigate whether this potential can be realized with learned reward models. We find that standard approaches fail in surprising ways, then develop a theoretical framework to diagnose the underlying causes.

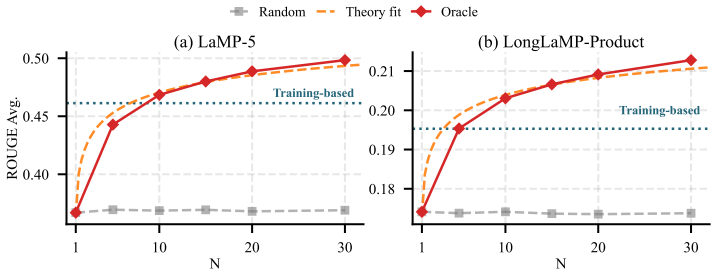

Figure 2: Oracle scaling on (a) LaMP-5 and (b) LongLaMP-Product. Oracle selection (red solid) closely follows the theoretical prediction $\bar{\mu} + \bar{\sigma} c\sqrt{\ln N}$ (orange dashed); the horizontal teal dotted line marks the per-user training-based baseline.

4.1 Standard Reward Models Fail to Scale

We evaluate the two reward model approaches defined in Section 2: Global RM trained on population-level data, and User-specific RM trained individually per user. Figure 3 presents the scaling curves on two representative tasks.

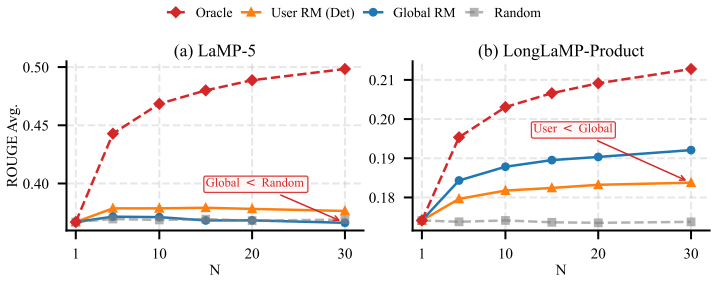

Figure 3: Scaling curves for standard reward models. (a) On LaMP-5, Global RM performs no better than random selection. (b) On LongLaMP-Product Review, User RM underperforms Global RM despite being explicitly trained on personalized user data. Both fall far short of the oracle upper bound.

Two unexpected phenomena emerge. First, on LaMP-5, Global RM achieves nearly identical performance to random selection; it fails to provide any meaningful signal for candidate selection. Second, on LongLaMP, User RM underperforms Global RM despite being explicitly personalized. This contradicts the intuition that user-specific training should improve personalization. Both approaches fall far short of the oracle upper bound, indicating that realizing the promise of TTP requires understanding why these standard methods fail.

4.2 From Correlation to Scaling

To analyze reward model quality, we introduce the correlation between learned and true rewards:

Definition 4.1 (Reward Model Correlation). For user u, the correlation between a learned reward model $\hat{r}_u$ and the true preference r_u^* is:

$$\rho_u = \mathrm{Corr}\left(\hat{r}_u(q,x),\, r_u^*(q,x)\right). \tag{3}$$

This correlation directly determines scaling behavior:

Lemma 4.2 (Correlation-Scaling Relationship). *Under uniformity assumption, for a reward model with correlation ρ_u, the Best-of-N utility satisfies:*

$$U_u(N) \approx \mu_u + \rho_u \cdot \sigma_u \cdot c\sqrt{\ln N} \tag{4}$$

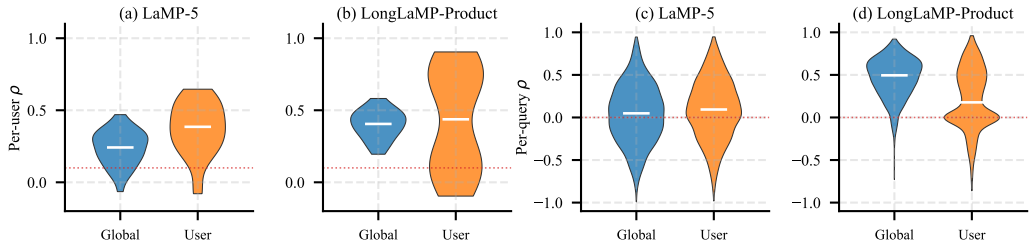

Figure 4: Correlation diagnostics on LaMP-5 and LongLaMP-Product Review. **(a, b)** Per-user correlation ρ_u: User RM achieves higher mean correlation but exhibits a heavy lower tail crossing the red dotted line at $\rho_u = 0.1$ (user-level collapse threshold); Global RM remains tightly concentrated. **(c, d)** Per-query correlation ρ_q: a substantial fraction of queries lie below the red dotted line at $\rho_q = 0$ (query-level reward hacking).

The proof is provided in Appendix A.2, and the bivariate-linearity assumption it relies on is empirically verified in Appendix C.2. Intuitively, correlation acts as a scaling coefficient: positive ρ yields increasing curves, $\rho \approx 0$ yields flat curves, and negative ρ yields decreasing curves. This framework allows us to diagnose reward model failures by examining correlation distributions.

4.3 Diagnosing Failure Modes

Lemma 4.2 motivates two diagnostic correlations: per-user ρ_u (averaged over queries) and per-query ρ_q (across candidates within a query). Figure 4 reveals two failure modes, each explaining one phenomenon in Figure 3. **Query-level reward hacking** ($\rho_q < 0$): on LaMP-5, 46% of queries exhibit negative correlation under Global RM (37% under User RM), causing it to systematically prefer worse candidates; this explains why Global RM cannot beat random in Figure 3(a). **User-level collapse** ($\rho_u < 0.1$): User RM achieves higher mean correlation than Global RM (0.37 vs 0.24 on LaMP-5; 0.43 vs 0.40 on LongLaMP-Product), but 25% of LongLaMP-Product users collapse under User RM (vs 0% under Global RM), where the reward model emits near-constant predictions and provides no useful signal; this explains why User RM falls below Global RM in Figure 3(b).

4.4 Unified Scaling Law

We formalize these failure modes and derive their joint effect on scaling behavior:

Definition 4.3 (**Failure Rates**). For a reward modeling approach, we define:

- **Collapse rate** α: Fraction of users with $\rho_u < 0.1$;
- **Reward hacking rate** β: Fraction of queries with $\rho_q < 0$.

Proposition 4.4 (**Unified Scaling Law**). *The population-level Best-of-N utility can be approximated as:*

$$\bar{U}(N) \approx \bar{\mu} + \underbrace{(1-\alpha)}_{\textit{non-collapsed}} \cdot \underbrace{[(1-\beta)\bar{\rho}_+ - \beta|\bar{\rho}_-|]}_{\textit{effective correlation}} \cdot \bar{\sigma} \cdot c\sqrt{\ln N} \tag{5}$$

where $\bar{\rho}_+ > 0$ is the mean correlation on non-hacked queries and $\bar{\rho}_- < 0$ is the mean correlation on hacked queries.

This formula reveals the **personalization-stability trade-off**: Global RM achieves stability (low α and β) by averaging across users, but this same averaging limits its ability to capture individual preferences. User RM achieves higher correlation for many users but suffers from both collapse and reward hacking due to limited per-user data.

Proposition 4.4 (proof in Appendix A.3) suggests that an effective approach must simultaneously achieve: (1) high base correlation through personalization, (2) low collapse rate for user-level stability, and (3) low reward hacking rate for query-level stability. We empirically validate the formula in Section 6.2 and Appendix C.1: the four measured quantities forecast observed Best-of-N scaling curves at relMAE below 3% across all tasks. In the next section, we show that probabilistic reward modeling is the unique design that achieves the three desiderata above.

5 Probabilistic Reward Modeling

The analysis in Section 4 reveals that effective TTP requires high correlation, low collapse rate α, and low reward hacking rate β. We now show that probabilistic reward modeling achieves all three by introducing a learned variance σ^2 that absorbs uncertainty rather than forcing premature commitment.

5.1 Theoretical Analysis

Consider the Gaussian negative log-likelihood (NLL) loss

$$\mathcal{L}_{\text{NLL}} = \tfrac{1}{2}\log\sigma^2(x) + \frac{(y-\mu(x))^2}{2\sigma^2(x)}, \tag{6}$$

whose gradient with respect to μ is $(\mu - y)/\sigma^2$. The learned σ^2 thus acts as an adaptive per-input weight, giving two complementary effects.

Lemma 5.1 (Gradient Buffering)**.** *For inputs where y is poorly predictable from x (collapse-prone users), increasing σ^2 attenuates the gradient on μ, preventing the constant-prediction equilibrium that traps deterministic models. This reduces the collapse rate: $\alpha^{prob} < \alpha^{det}$.*

Lemma 5.2 (Implicit Regularization)**.** *For inputs with inconsistent training signals, NLL drives σ^2 large, down-weighting their contribution to the gradient on μ. This prevents overfitting to spurious patterns that produce negative per-query correlation, reducing the hacking rate: $\beta^{prob} < \beta^{det}$.*

Together (proofs in Appendix A.4), the two lemmas give probabilistic User RM the unique combination required by Proposition 4.4: low α, low β, and high $\bar{\rho}_+$. This produces the monotonic Best-of-N scaling we observe empirically (Section 6.2), in contrast to the flat or non-monotonic curves of deterministic models when failure modes dominate.

5.2 Implementation

We extend a language model backbone with two heads predicting per-input mean and variance:

$$\mu(x) = \text{Sigmoid}(\text{MLP}_\mu(\mathbf{h})), \qquad \sigma^2(x) = \text{Softplus}(\text{MLP}_\sigma(\mathbf{h})), \tag{7}$$

where $\mathbf{h}$ is the masked-mean-pooled hidden state from the backbone, which we adapt per user via LoRA for parameter efficiency.

The training objective combines the NLL loss in Eq. (6) with a margin loss that sharpens ranking among high-quality candidates, the regime that matters for Best-of-N selection:

$$\mathcal{L}_{\text{contrast}} = \sum_{\substack{i,j:\, y_i > y_j \\ y_i > \tau}} \max(0, m - (\mu_i - \mu_j)), \tag{8}$$

where τ is a high-score threshold and m is the margin, giving $\mathcal{L} = \mathcal{L}_{\text{NLL}} + \lambda\mathcal{L}_{\text{contrast}}$. At inference, we score candidates by μ alone; the variance is used only at training time. Backbone, hyperparameters, and prompt templates appear in Appendix E, an ablation across architectural components and the loss decomposition is in Appendix B.3, and computational-cost measurements (training and inference timing) are in Appendix B.7.

6 Experiments

We evaluate Test-Time Personalization on five personalized text-generation tasks, validating both the predictive scaling law of Proposition 4.4 and the practical effectiveness of probabilistic RM.

6.1 Setup

Datasets. We use five tasks from LaMP [3] (4-News Headline Generation, 5-Scholarly Title Generation) and LongLaMP [13] (Abstract Generation, Product Review Writing, Topic Writing). Following Tan et al. [4], we select the top-30 users with the longest profiles for LaMP and the top-20 for LongLaMP.

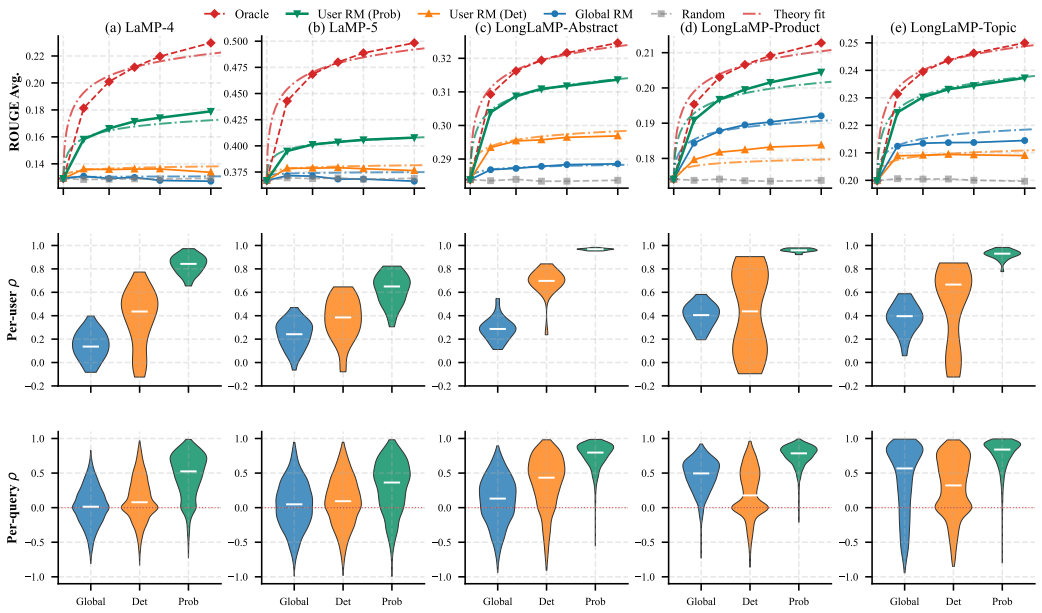

Figure 5: Main TTP results under the RAG policy across five tasks (a)–(e). **Top row**: observed Best-of-N ROUGE (solid lines with markers) and theory predictions (dash-dot lines), where each method's prediction is computed by substituting its four diagnostic quantities $(\alpha, \beta, \bar{\rho}_+, \bar{\rho}_-)$ into Proposition 4.4 (oracle uses Theorem 3.1); theory and observation agree at relMAE $< 3\%$. **Middle row**: per-user correlation ρ_u violin distributions. **Bottom row**: per-query correlation ρ_q violin distributions; the red dotted line marks $\rho_q = 0$. Probabilistic User RM (green) is the only method whose curve scales monotonically, and the right-shifted, tightened correlation distributions confirm that this gain is driven by the elimination of user-level collapse and query-level reward hacking.

Policy Families. Our main results use a retrieval-augmented generation (RAG) policy [3] that conditions on user-history examples retrieved per query. To probe whether TTP gains depend on the choice of generator, we additionally evaluate two alternatives in Appendix B.1: *Persona prompting* [14] (a user-style description synthesised from 15 history samples by an external LLM) and *Persona+RAG* (concatenating both). All three use `Qwen3-4B-Instruct` [15] as the generator and produce N=30 candidates per query at temperature T=1.7 (LaMP) or T=1.5 (LongLaMP). We use 80% of each user's data to train reward models and 20% for evaluation.

Reward Models. We compare five selection strategies: **Random** (uniform pick), **Global RM** (single model on pooled data), **Deterministic User RM** (per-user, MSE loss with high-score contrastive loss), **Probabilistic User RM** (per-user, Gaussian NLL with high-score contrastive loss; ours), and **Oracle** (selection by ground-truth ROUGE). All reward models share the `Qwen2.5-1.5B-Instruct` [16] backbone with LoRA (r=8); the probabilistic variant adds a soft-plus variance head. For comparison with training-based personalization, we also fine-tune per-user LoRA adapters on the policy following [4]. Variance-based inference strategies are explored in Appendix B.6 but offer no improvement over mean-based selection.

Evaluation. We sample $N \in \{1, 5, 10, 15, 20, 30\}$ candidates per validation query and report the average of ROUGE-1 and ROUGE-L across users. Each experiment is repeated 10 times; standard deviations remain below 0.007 across methods and N. Robustness to ROUGE is assessed with BERTScore and an LLM-as-judge in Appendix B.2; sensitivity to per-user data volume, LoRA rank, and statistical stability across repetitions are reported in Appendix B.5.

6.2 Main Results

Figure 5 presents three views of evaluation with RAG policy: scaling curves with theory overlay, per-user correlation distributions, and per-query correlation distributions. We highlight three findings.

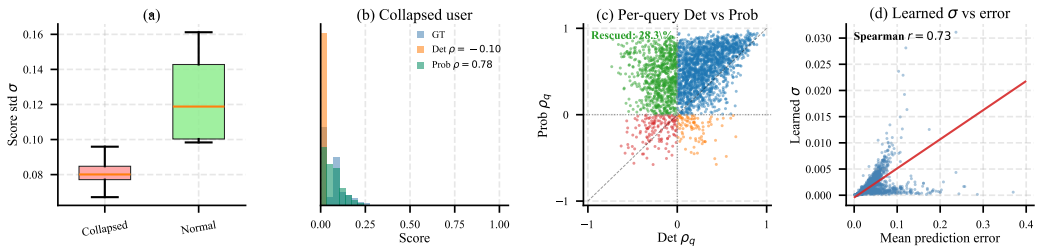

Figure 6: Failure-mode analysis on LaMP-4. **(a)** Ground-truth ROUGE standard deviation distinguishes collapsed users ($\rho_u < 0.1$ under Det RM) from normal users ($\rho_u > 0.5$). **(b)** Score histogram of a representative collapsed user: GT scores cluster near zero, Det RM degenerates to near-constant predictions ($\rho = -0.10$), while Prob RM preserves meaningful variation ($\rho = 0.78$). **(c)** Per-query correlation scatter (Det vs Prob): green points in the upper-left quadrant are queries where Prob "rescues" Det's negative correlation. **(d)** Learned variance σ^2 correlates strongly with prediction error (Spearman $r = 0.73$), evidence that Prob RM is well-calibrated.

Probabilistic RM is the only method that scales reliably. Across all five tasks (Figure 5, top row), Probabilistic User RM is the only method whose Best-of-N curve increases monotonically. By N=30, Probabilistic RM closes between 51% and 84% of the oracle gap depending on the task, and consistently surpasses the per-user LoRA training-based baseline by N=5–10 on three of the five tasks (per-task numbers in Appendix B.1). The same selector ranking holds under two additional policy families (Persona prompting and Persona+RAG, Appendix B.1), confirming that TTP behaves as a policy-orthogonal amplifier whose effect is determined by the reward model rather than the underlying generator.

Theory predictions match observations and the correlation distributions show why. The dash-dot theory curves in the top row are computed by substituting each reward model's four measured quantities $(\alpha, \beta, \bar{\rho}_+, \bar{\rho}_-)$ into Proposition 4.4, with no per-method fitting; they yield effective correlations $\rho_{\text{eff}} = 0.20/0.19/0.61$ for Global / Det / Prob RM and predict the observed curves at relMAE $4.4\%/2.7\%/1.7\%$ averaged across the five tasks (per-task table in Appendix C.1). The middle and bottom rows of Figure 5 explain why these ρ_{eff} values arise: probabilistic modeling shifts both per-user and per-query correlation distributions rightward and tightens them, dropping the user-level collapse rate α from 0.15 (Det) to 0 and the query-level hacking rate β from 0.25 to 0.06 averaged across all five tasks. These distributional shifts are exactly the conditions Proposition 4.4 requires for monotonic improvement, completing a tight loop between theory, diagnosis, and method.

Task characteristics shape TTP headroom. Long-form tasks (Abstract, Product, Topic) yield larger oracle gaps and correspondingly larger TTP gains than short-form tasks (Headlines, Titles), where small token differences produce large ROUGE swings and reduce learnable signal. The diagnostic framework explains this directly: short-form tasks exhibit smaller Oracle $\bar{\sigma}$, hence a smaller logarithmic-growth coefficient $\bar{\sigma}c\sqrt{\ln N}$ (Theorem 3.1).

6.3 Failure-Mode Analysis

To understand *when* and *how* the two failure modes manifest in practice, we conduct a deeper analysis on LaMP-4 (Figure 6); cross-task results appear in Appendix C.3.

When does user-level collapse occur? Figure 6 (a) groups users by their ground-truth ROUGE variance: collapsed users (under Det RM) have significantly lower label variance than normal users ($p < 0.001$). A representative collapsed user (Figure 6 (b)) illustrates the mechanism: when GT scores cluster narrowly, Det RM emits near-constant predictions ($\rho = -0.10$), while Prob RM preserves meaningful variation ($\rho = 0.78$). This aligns with Lemma 5.1: NLL training increases σ^2 on weak-signal users, attenuating the gradient on μ and preventing the regression-to-the-mean that traps deterministic models. The collapse rate drops from $\alpha^{\text{det}} = 0.20$ to $\alpha^{\text{prob}} = 0$ on LaMP-4, and to 0 on every other task as well.

When does query-level hacking occur? Figure 6 (c) plots per-query Det vs Prob correlations. The green points (upper-left quadrant) are queries where Det produces negative correlation but Prob produces positive correlation, accounting for 28% of all LaMP-4 queries; combined with cases where both are positive (blue), Prob's negative-correlation rate drops from $\beta^{\text{det}} = 0.33$ to $\beta^{\text{prob}} = 0.08$. Figure 6 (d) explains why: Prob RM's learned σ^2 rises with prediction error (Spearman $r = 0.73$), so during training the loss naturally down-weights queries it cannot fit. This implicit regularization (Lemma 5.2) prevents Det RM's overfitting to spurious patterns on hard queries, which is what produces negative correlation in the first place.

6.4 Extrapolation Beyond the Main Setting

The main results use $N \leq 30$, data-rich users, and ROUGE as the quality signal. Figure 7 tests how far the conclusions extend along each axis; full tables are in Appendices D.1, D.2, and D.3.

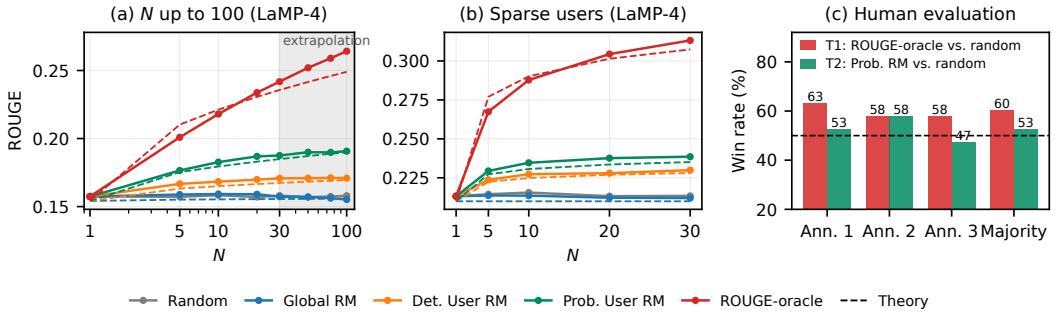

Figure 7: Extrapolation along three axes. **(a)** Best-of-N up to N=100 on a fresh LaMP-4 pool: observed (solid) vs. Proposition 4.4 with diagnostics measured at $N \leq 30$ (dashed; shaded region is extrapolation). **(b)** The same pipeline on 30 sparse LaMP-4 users with 31–100 history items. **(c)** Blind pairwise human evaluation: win rate of the selected output over a random candidate from the same pool, per annotator and by majority vote (dashed line: chance).

Larger candidate pools. On 100-candidate pools (Figure 7a; LongLaMP-Product in Appendix D.1), the three regimes predicted by Proposition 4.4 all appear: Probabilistic RM keeps rising, Global RM falls below random as its hacking is amplified by the larger pool, and Deterministic RM saturates; diagnostics measured only at $N \leq 30$ predict the N=50–100 curves at relMAE $\leq 2.1\%$. Since the marginal gain per candidate decays as $1/(N\sqrt{\ln N})$ against linear generation cost, $N \in [5, 30]$ captures most of the benefit.

Sparser users. On 30 LaMP-4 users with 31–100 history items instead of 500+ (Figure 7b), Probabilistic RM still gains $+11.8\%$ over random at N=30 with the same selector ranking, while the zero-shot Global RM drops to random; the reduced capture (25% of the oracle gap) is predicted by the law from the lower measured ρ_{eff}. Collapse does *not* spike (α^{prob}=0.03): these users have higher label variance, and by Lemma 5.1 it is low label variance rather than sparsity that creates the collapse attractor; sparsity instead raises hacking.

Other quality signals. Selecting by ROUGE improves BERTScore on every task ($+2.1\%$ to $+10.2\%$ at N=30), so the gains are not an artifact of the optimized metric. In a blind pairwise human evaluation (76 items, three annotators passing catch trials; Figure 7c), every annotator preferred the ROUGE-oracle-selected output over a random candidate (majority 61%, one-sided p=0.13), while the Prob-RM-versus-random test is near chance because same-pool candidates typically differ by only a few words; the sample is small, so we read it as supporting rather than establishing alignment.

7 Related Work

LLM Personalization. Existing approaches to LLM personalization fall into three families. *Prompting-based methods* augment the input context with user information: LaMP [3] retrieves user history via dense retrieval, while PAG [2] first generates user profiles with LLMs to enrich personalization. *Adaptation-based methods* fine-tune model parameters per user: OPPU [4] introduces per-user

LoRA adapters, and FDLoRA [17] improves efficiency through federated learning. *Alignment-based methods* formulate personalization as multi-objective optimization: Rewarded Soup [18] interpolates separately trained policy weights to balance preference dimensions. All three modify what the model sees or how it is trained; we instead scale the policy output space at inference time and study when selection-based scaling is reliable.

Uncertainty in Reward Modeling. Uncertainty quantification has been used to mitigate reward hacking in general alignment. Ensemble approaches train multiple reward models and use disagreement as the uncertainty signal [19]. UP-RLHF [20] trains diverse LoRA ensembles with uncertainty-penalized optimization, while PURM [21] extends Bradley–Terry to a distributional reward. A common feature of these works is that uncertainty is consumed at *inference time*, e.g., as a PPO penalty or a selection adjustment. Our use of the variance head is structurally different: σ^2 acts purely at *training time* via gradient buffering and implicit regularization (Lemmas 5.1, 5.2); inference uses only the predicted mean. This shift exposes a personalization-specific role for probabilistic modeling that prior alignment-focused work does not address.

Test-Time Scaling for Personalization. Recent work has shown that scaling test-time compute can improve LLM alignment, but the available compute can be allocated to several distinct targets. Three are particularly relevant for personalization: *(i) scaling user interactions*: T-POP [11] learns per-user preferences online via dueling bandits, requiring repeated preference queries; *(ii) scaling reward-model reasoning*: generative reward models unroll chain-of-thought tokens inside the reward model itself, trading inference cost for ranking accuracy; and *(iii) scaling policy outputs*: sampling N candidates from a personalized policy and selecting the best with a per-user reward model, the axis we study here. The three axes are complementary rather than competing; our diagnostic framework (Section 4) applies to any reward model used in axis (iii) and could be combined with (i) or (ii). Training-free LLM-as-judge methods, such as persona-conditioned pairwise tournaments, instantiate axis (ii) with a general-purpose LLM as the personalized reward model, but achieve only coin-flip accuracy on per-user style matching in our setting (Appendix B.4), confirming that per-user trained reward models remain necessary for stylistic personalization. TPO [22] iteratively refines outputs through textual critiques but does not target personalization.

8 Conclusion and Discussion

Summary. We studied **Test-Time Personalization** (TTP) along a new axis by scaling policy outputs at inference for personalized text generation. Our central contribution is a *predictive scaling law* that links four measurable reward-model properties to the Best-of-N utility curve, validated empirically at relMAE $< 3\%$. The law identifies two failure modes (user-level collapse and query-level reward hacking), and we prove that probabilistic reward modeling overcomes both via training-time gradient buffering and implicit regularization. The resulting probabilistic User RM is the only method whose Best-of-N curve scales monotonically across all 15 (task, policy) cells we tested, confirming TTP as a policy-orthogonal amplifier.

Limitations. Our evaluation focuses on *personalized text generation* with publicly available benchmarks (LaMP, LongLaMP) using ROUGE-derived ground-truth rewards. The oracle gap therefore measures headroom under this proxy, not under human preference; BERTScore transfer and a small human evaluation point the same way (Section 6.4), but a larger human study is needed to quantify how much of the gap is perceived personalization. Generalization beyond text generation (e.g., dialogue, agentic task) remains untested. The framework also assumes per-user data sufficient to train a small probabilistic reward model; cold-start scenarios fall outside the regime we study and are better addressed by orthogonal axes such as scaling user interactions (Section 7). Finally, training separate reward models per user may pose deployment challenges at scale, motivating future work on parameter-efficient cross-user sharing.

Future Directions. The diagnostic framework opens several avenues. First, the three test-time axes (user interactions, reward-model reasoning, policy outputs) are complementary; combining probabilistic per-user RMs with generative reward modeling is a natural next step. Second, extending probabilistic reward modeling to non-text modalities would test whether the gradient-buffering and implicit-regularization mechanisms transfer to settings with richer label structure.

Acknowledgments and Disclosure of Funding

This work was supported in part by the UK Engineering and Physical Sciences Research Council (EPSRC) through the Prosperity Partnership scheme (grant no. UKRI566) and a Turing AI Fellowship (grant no. EP/V020579/1, EP/V020579/2). The authors acknowledge use of the research computing facility at King's College London, King's Computational Research, Engineering and Technology Environment (CREATE).

References

[1] Jiahong Liu, Zexuan Qiu, Zhongyang Li, Quanyu Dai, Wenhao Yu, Jieming Zhu, Minda Hu, Menglin Yang, Tat-Seng Chua, and Irwin King. A survey of personalized large language models: Progress and future directions, 2025. URL https://arxiv.org/abs/2502.11528.

[2] Chris Richardson, Yao Zhang, Kellen Gillespie, Sudipta Kar, Arshdeep Singh, Zeynab Raeesy, Omar Zia Khan, and Abhinav Sethy. Integrating summarization and retrieval for enhanced personalization via large language models, 2023. URL https://arxiv.org/abs/2310.20081.

[3] Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. LaMP: When large language models meet personalization. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7370–7392, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.399. URL https://aclanthology.org/2024.acl-long.399/.

[4] Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. Democratizing large language models via personalized parameter-efficient fine-tuning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6476–6491, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.372. URL https://aclanthology.org/2024.emnlp-main.372/.

[5] Linhai Zhang, Jialong Wu, Deyu Zhou, and Yulan He. PROPER: A progressive learning framework for personalized large language models with group-level adaptation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16399–16411, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.800. URL https://aclanthology.org/2025.acl-long.800/.

[6] Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. In *Adaptive Foundation Models: Evolving AI for Personalized and Efficient Learning*, 2024. URL https://openreview.net/forum?id=EMrnoPRvxe.

[7] Stéphane Aroca-Ouellette, Natalie Mackraz, Barry-John Theobald, and Katherine Metcalf. Aligning LLMs by predicting preferences from user writing samples. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=eUMGCipgtE.

[8] Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for LLM problem-solving. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=VNckp7JEHn.

[9] Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

[10] Junhong Shen, Hao Bai, Lunjun Zhang, Yifei Zhou, Amrith Setlur, Shengbang Tong, Diego Caples, Nan Jiang, Tong Zhang, Ameet Talwalkar, and Aviral Kumar. Thinking vs. doing: Agents that reason by scaling test-time interaction. In *Workshop on Scaling Environments for Agents*, 2025. URL https://openreview.net/forum?id=uhigrPHBm5.

[11] Zikun Qu, Min Zhang, Mingze Kong, Xiang Li, Zhiwei Shang, Zhiyong Wang, Yikun Ban, Shuang Qiu, Yao Shu, and Zhongxiang Dai. T-pop: Test-time personalization with online preference feedback, 2025. URL https://arxiv.org/abs/2509.24696.

[12] Pinyi Zhang, Ting-En Lin, Yuchuan Wu, Jingyang Chen, Zongqi Wang, Hua Yang, Xu Ze, Fei Huang, Yongbin Li, and Kai Zhang. P-genRM: Personalized generative reward model with test-time user-based scaling. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=hXNApWLBZG.

[13] Ishita Kumar, Snigdha Viswanathan, Sushrita Yerra, Alireza Salemi, Ryan A. Rossi, Franck Dernoncourt, Hanieh Deilamsalehy, Xiang Chen, Ruiyi Zhang, Shubham Agarwal, Nedim Lipka, Chien Van Nguyen, Thien Huu Nguyen, and Hamed Zamani. Longlamp: A benchmark for personalized long-form text generation, 2024. URL https://arxiv.org/abs/2407.11016.

[14] Michael J. Ryan, Omar Shaikh, Aditri Bhagirath, Daniel Frees, William Held, and Diyi Yang. SynthesizeMe! inducing persona-guided prompts for personalized reward models in LLMs. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8045–8078, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.397. URL https://aclanthology.org/2025.acl-long.397/.

[15] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

[16] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

[17] Jiaxing Qi, Zhongzhi Luan, Shaohan Huang, Carol Fung, Hailong Yang, and Depei Qian. Fdlora: Personalized federated learning of large language model via dual lora tuning. *arXiv preprint arXiv:2406.07925*, 2024.

[18] Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=lSbbC2VyCu.

[19] Xingzhou Lou, Dong Yan, Wei Shen, Yuzi Yan, Jian Xie, and Junge Zhang. Uncertainty-aware reward model: Teaching reward models to know what is unknown, 2025. URL https://arxiv.org/abs/2410.00847.

[20] Yuanzhao Zhai, Yu Lei, Han Zhang, Yue Yu, Kele Xu, Dawei Feng, Bo Ding, and Huaimin Wang. Uncertainty-penalized reinforcement learning from human feedback with diversified reward lora ensembles. *Information Processing and Management*, 63(3):104548, 2026. ISSN 0306-4573. doi: https://doi.org/10.1016/j.ipm.2025.104548. URL https://www.sciencedirect.com/science/article/pii/S0306457325004893.

[21] Wangtao Sun, Xiang Cheng, Xing Yu, Haotian Xu, Zhao Yang, Shizhu He, Jun Zhao, and Kang Liu. Probabilistic uncertain reward model, 2025. URL https://arxiv.org/abs/2503.22480.

[22] Yafu Li, Xuyang Hu, Xiaoye Qu, Linjie Li, and Yu Cheng. Test-time preference optimization: On-the-fly alignment via iterative textual feedback, 2025. URL https://arxiv.org/abs/2501.12895.

[23] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

[24] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

[25] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mZn2Xyh9Ec.

[26] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

A Theoretical Proofs

In this section, we provide formal proofs or theoretical analysis for all theoretical results presented in the main text. Our theoretical framework rests on a chain of results that progressively characterizes test-time personalization:

1. **Theorem 3.1 (Oracle Scaling Law, Section A.1)**: Establishes the theoretical ceiling for TTP, expected utility grows as $O(\sqrt{\ln N})$ with oracle selection.
2. **Lemma 4.2 (Correlation-Scaling Relationship, Section A.2)**: Shows that reward model correlation directly determines scaling behavior, providing a diagnostic tool for analyzing RM quality.
3. **Proposition 4.4 (Unified Scaling Law, Section A.3)**: Derives how the two failure modes, collapse rate α and hacking rate β, jointly determine population-level scaling.
4. **Lemmas 5.1 and 5.2 (Gradient Buffering & Implicit Regularization, Section A.4)**: Explains the mechanisms by which probabilistic reward modeling reduces both failure modes.

Throughout the proofs, we introduce necessary assumptions and provide remarks connecting theoretical insights to empirical observations. Table A1 summarizes the key assumptions used in our analysis.

Table A1: Summary of assumptions used in theoretical analysis.

Assumption	Used In	Description
Sub-Gaussian rewards	Theorem 3.1	Reward distributions have light tails
Correlation uniformity	Lemma 4.2	Local correlation $\approx$ global correlation
Collapse absorbs hacking	Proposition 4.4	β measured on non-collapsed users
Variance homogeneity	Proposition 4.4	σ variation is second-order

A.1 Proof of Theorem 3.1

We first introduce the sub-Gaussian property and a key lemma on the maximum of random variables, then prove Theorem 3.1.

Definition A.1 (Sub-Gaussian Random Variable)**.** A random variable X with mean $\mu = \mathbb{E}[X]$ is σ-sub-Gaussian if:

$$\mathbb{E}\left[\exp\left(\lambda(X-\mu)\right)\right] \leq \exp\left(\frac{\lambda^2\sigma^2}{2}\right), \quad \forall\lambda \in \mathbb{R} \tag{9}$$

where $\sigma > 0$ is the sub-Gaussian parameter. This assumption covers a wide range of bounded or light-tailed distributions, including typical LLM quality scores.

Lemma A.2 (Upper Bound on Expected Maximum)**.** *Let $X_1, \ldots, X_N$ be i.i.d. σ-sub-Gaussian random variables with mean μ. Then the expected maximum is bounded by:*

$$\mathbb{E}\left[\max_{i\in[N]} X_i\right] \leq \mu + \sigma\sqrt{2\ln N} \tag{10}$$

Proof. For any $\lambda > 0$, using Jensen's inequality and the monotonicity of the exponential function:

$$\exp\left(\lambda\mathbb{E}\left[\max_i(X_i-\mu)\right]\right) \leq \mathbb{E}\left[\exp\left(\lambda\max_i(X_i-\mu)\right)\right] = \mathbb{E}\left[\max_i \exp(\lambda(X_i-\mu))\right] \tag{11}$$

Bounding the maximum by the sum:

$$\mathbb{E}\left[\max_i \exp(\lambda(X_i-\mu))\right] \leq \sum_{i=1}^{N}\mathbb{E}\left[\exp(\lambda(X_i-\mu))\right] \tag{12}$$

Applying the sub-Gaussian property (Definition A.1):

$$\sum_{i=1}^{N}\mathbb{E}\left[\exp(\lambda(X_i-\mu))\right] \leq N\exp\left(\frac{\lambda^2\sigma^2}{2}\right) \tag{13}$$

Taking logarithms on both sides:

$$\lambda \mathbb{E}\left[\max_i (X_i - \mu)\right] \leq \ln N + \frac{\lambda^2 \sigma^2}{2} \implies \mathbb{E}\left[\max_i X_i\right] \leq \mu + \frac{\ln N}{\lambda} + \frac{\lambda \sigma^2}{2} \tag{14}$$

The right-hand side is minimized when $\lambda = \sqrt{2 \ln N}/\sigma$. Substituting this value yields the bound $\mu + \sigma\sqrt{2 \ln N}$. □

Proof. For a user u with query q, assuming the true rewards $\{r_u^*(q, x_i)\}_{i=1}^N$ are i.i.d. σ_u-sub-Gaussian with mean μ_u, we apply Lemma A.2:

$$U_{\text{oracle},u}(N) = \mathbb{E}\left[\max_{i \in [N]} r_u^*(q, x_i)\right] \leq \mu_u + c \cdot \sigma_u \cdot \sqrt{\ln N} \tag{15}$$

where $c = \sqrt{2}$ is the theoretical scaling coefficient derived from the sub-Gaussian bound.

Taking the expectation over the population of users:

$$\bar{U}_{\text{oracle}}(N) = \mathbb{E}_u[U_{\text{oracle},u}(N)] \leq \mathbb{E}_u[\mu_u] + c\sqrt{\ln N} \cdot \mathbb{E}_u[\sigma_u] = \bar{\mu} + c\bar{\sigma}\sqrt{\ln N} \tag{16}$$

Thus, the oracle utility is upper bounded by a logarithmic growth term scaled by the average reward noise $\bar{\sigma}$. □

Remark on Bound Tightness. While Theorem 3.1 provides an upper bound with $c = \sqrt{2}$, our empirical observations suggest that the scaling curve tightly follows this boundary. This indicates that the reward distributions in practice are not merely sub-Gaussian but exhibit tail behaviors close to the Gaussian limit (where the bound becomes an asymptotic equality). Consequently, we treat c as an effective constant in our analysis to absorb minor distributional deviations, recognizing that $c \approx \sqrt{2}$ represents the fully saturated theoretical potential.

A.2 Proof of Lemma 4.2

This lemma establishes how reward model correlation determines TTP scaling behavior. We first present the idealized result under a uniformity assumption, then discuss when this assumption breaks down in practice.

Setup and Notation For a user u, let $r^* = r_u^*(q, x)$ denote the true reward and $\hat{r} = \hat{r}_u(q, x)$ denote the predicted reward for a query-response pair. We assume both are standardized to have mean μ and consider their correlation:

$$\rho_u = \text{Corr}(\hat{r}, r^*) = \frac{\mathbb{E}[(\hat{r} - \mu)(r^* - \mu)]}{\sigma_{\hat{r}} \sigma_{r^*}} \tag{17}$$

Assumption A.3 (Correlation Uniformity)**.** The correlation ρ_u between predicted and true rewards is approximately uniform across the score distribution. Formally, for any subset $\mathcal{S} \subseteq [0, 1]$ of the score range:

$$\rho_u^{(\mathcal{S})} := \text{Corr}(\hat{r}, r^* \mid r^* \in \mathcal{S}) \approx \rho_u \tag{18}$$

This assumption is analogous to homoscedasticity in regression: the predictive relationship is stable across different regions of the target distribution.

Lemma A.4 (Correlation-Scaling Relationship, Restated)**.** *Under Assumption A.3, for a reward model with correlation ρ_u, the Best-of-N utility satisfies:*

$$U_u(N) \approx \mu_u + \rho_u \cdot \sigma_u \cdot c\sqrt{\ln N} \tag{19}$$

where $c > 0$ is the same constant as in Theorem 3.1.

Proof. We assume the joint distribution of predicted and true rewards follows a Bivariate Normal distribution (or satisfies the property of linear conditional expectation). Under joint normality (or linear regression structure), the conditional expectation of true reward given predicted reward is:

$$\mathbb{E}[r^* \mid \hat{r}] = \mu + \rho_u \cdot \frac{\sigma_{r^*}}{\sigma_{\hat{r}}}(\hat{r} - \mu) \tag{20}$$

Let $\hat{x}^* = \arg\max_{i\in[N]} \hat{r}(x_i)$ be the candidate selected by the reward model. The expected true reward of this selection is:

$$\mathbb{E}[r^*(\hat{x}^*)] = \mathbb{E}\left[\mathbb{E}[r^* \mid \hat{r}(\hat{x}^*)]\right] = \mu + \rho_u \cdot \frac{\sigma_{r^*}}{\sigma_{\hat{r}}} \cdot \mathbb{E}[\hat{r}(\hat{x}^*) - \mu] \tag{21}$$

Since $\hat{r}(x_i)$ are i.i.d. sub-Gaussian, by Lemma A.2:

$$\mathbb{E}[\hat{r}(\hat{x}^*)] = \mathbb{E}\left[\max_{i\in[N]} \hat{r}(x_i)\right] = \mu + c \cdot \sigma_{\hat{r}} \cdot \sqrt{\ln N} \tag{22}$$

Substituting into the previous equation:

$$U_u(N) = \mathbb{E}[r^*(\hat{x}^*)] = \mu + \rho_u \cdot \frac{\sigma_{r^*}}{\sigma_{\hat{r}}} \cdot c \cdot \sigma_{\hat{r}} \cdot \sqrt{\ln N} = \mu_u + \rho_u \cdot \sigma_u \cdot c\sqrt{\ln N} \tag{23}$$

where $\sigma_u := \sigma_{r^*}$ for notational consistency. □

Remark. In practice, Assumption A.3 often does not hold exactly. High-score regions are typically harder to model due to data sparsity, subtle quality distinctions, and regression to the mean, leading to $\rho_u^{(\text{high})} < \rho_u^{(\text{overall})}$. Since Best-of-N selection operates primarily in the right tail of the score distribution, the effective correlation for TTP is lower than the overall correlation. This explains why in our main experiments (Figure 5), even when probabilistic RM achieves high user-level correlation on LongLaMP tasks, a gap remains between TTP performance and the oracle upper bound.

A.3 Proof of Proposition 4.4

This proposition establishes how the two failure modes, user-level collapse and query-level reward hacking, jointly determine TTP scaling behavior.

Definitions and Assumptions. We first formalize the failure rates introduced in Definition 4.3:

Definition A.5 (Failure Rates, Restated)**.** For a reward modeling approach:

- **Collapse rate** α: Fraction of users with per-user correlation $\rho_u < \tau_c$, where $\tau_c = 0.1$ is the collapse threshold.
- **Reward hacking rate** β: Fraction of queries (among non-collapsed users) with per-query correlation $\rho_q < 0$.

Assumption A.6 (Collapse Absorbs Hacking). For collapsed users ($\rho_u < \tau_c$), the reward model outputs near-constant predictions, rendering query-level correlation undefined or negligible. Thus, the hacking rate β is measured only among non-collapsed users.

This assumption reflects the intuition that collapse represents a more severe failure mode: when a reward model cannot discriminate between candidates at all, the notion of "selecting worse candidates" (hacking) becomes moot.

Assumption A.7 (Variance Homogeneity). The variance σ_u of true rewards is approximately constant across users and query types. Variations in σ across collapsed/non-collapsed users or hacked/non-hacked queries are treated as second-order effects.

Proposition A.8 (Unified Scaling Law, Restated)**.** *Under Assumptions A.3, A.6, and A.7, the population-level Best-of-N utility can be approximated as:*

$$\bar{U}(N) \approx \bar{\mu} + (1-\alpha) \cdot [(1-\beta)\bar{\rho}_+ - \beta|\bar{\rho}_-|] \cdot \bar{\sigma} \cdot c\sqrt{\ln N} \tag{24}$$

where $\bar{\rho}_+ > 0$ is the mean correlation on non-hacked queries and $\bar{\rho}_- < 0$ is the mean correlation on hacked queries (both measured among non-collapsed users).

Proof. We derive the result through a two-level decomposition. Partition users into collapsed (fraction α) and non-collapsed (fraction $1-\alpha$):

$$\bar{U}(N) = \alpha \cdot \bar{U}_{\text{collapsed}}(N) + (1-\alpha) \cdot \bar{U}_{\text{non-collapsed}}(N) \tag{25}$$

For collapsed users, $\rho_u \approx 0$, so by Lemma 4.2, $\bar{U}_{\text{collapsed}}(N) \approx \bar{\mu}$.

Among non-collapsed users, partition queries into hacked (fraction β) and non-hacked (fraction $1-\beta$). Applying Lemma 4.2 to each partition:

$$\bar{U}_{\text{non-hacked}}(N) \approx \bar{\mu} + \bar{\rho}_+ \cdot \bar{\sigma} \cdot c\sqrt{\ln N} \tag{26}$$

$$\bar{U}_{\text{hacked}}(N) \approx \bar{\mu} + \bar{\rho}_- \cdot \bar{\sigma} \cdot c\sqrt{\ln N} \tag{27}$$

Combining the query-level results:

$$\bar{U}_{\text{non-collapsed}}(N) = \bar{\mu} + [(1-\beta)\bar{\rho}_+ - \beta|\bar{\rho}_-|] \cdot \bar{\sigma} \cdot c\sqrt{\ln N} \tag{28}$$

Substituting back into the user-level decomposition:

$$\bar{U}(N) = \bar{\mu} + (1-\alpha) \cdot [(1-\beta)\bar{\rho}_+ - \beta|\bar{\rho}_-|] \cdot \bar{\sigma} \cdot c\sqrt{\ln N} \tag{29}$$

□

Remark on the trade-off. Proposition A.8 reveals a fundamental trade-off: Global RM achieves stability (low α, low β) but weak personalization (low $\bar{\rho}_+$); deterministic User RM achieves high $\bar{\rho}_+$ but suffers from instability (high α or β); probabilistic User RM achieves both high correlation and stability through gradient buffering and implicit regularization (Lemmas 5.1 and 5.2), enabling effective TTP. The unified scaling law thus provides a diagnostic framework for predicting TTP performance by measuring α, β, $\bar{\rho}_+$, and $\bar{\rho}_-$.

Remark on the collapsed-user assumption. Assumption A.6 treats the contribution of collapsed users as negligible because their per-user correlation is approximately zero. In our data, however, collapsed users tend to have a small but non-zero negative correlation (e.g., $\bar{\rho}_{\text{collapsed}} \approx -0.027$ on LaMP-4 and -0.030 on LongLaMP-Product Review). Replacing the $\rho_u \approx 0$ assumption with the empirically measured $\bar{\rho}_{\text{collapsed}}$ yields a refined formula

$$\bar{U}(N) \approx \bar{\mu} + \left[(1-\alpha)\rho_{\text{eff}} + \alpha\bar{\rho}_{\text{collapsed}}\right] \cdot \bar{\sigma} \cdot c\sqrt{\ln N}. \tag{30}$$

The refinement is small in magnitude (improving relMAE on LaMP-4 Det RM from 9.1% to 8.8%, similarly on the other tasks where collapsed users are most prevalent) and *strengthens* the main conclusion: collapsed users contribute slightly *negative* scaling, so probabilistic modeling, which eliminates collapse altogether (α^{prob}=0), is even more important than the original analysis suggests.

A.4 Theoretical Analysis of Lemma 5.1 and Lemma 5.2

In this section, we provide theoretical analysis of how probabilistic reward modeling addresses the two failure modes identified in Section 4. Both mechanisms operate during training by leveraging the learned variance σ^2 to modulate gradient flow and sample weighting.

Preliminaries: NLL vs MSE Training. Consider a reward model that predicts mean $\mu(x)$ and variance $\sigma^2(x)$ for input x with ground-truth label y. The two training objectives are:

MSE Loss:

$$\mathcal{L}_{\text{MSE}} = (y-\mu)^2 \tag{31}$$

Gaussian NLL Loss:

$$\mathcal{L}_{\text{NLL}} = \frac{1}{2}\log\sigma^2 + \frac{(y-\mu)^2}{2\sigma^2} \tag{32}$$

The key difference lies in their gradients with respect to μ:

$$\frac{\partial\mathcal{L}_{\text{MSE}}}{\partial\mu} = -2(y-\mu) \tag{33}$$

$$\frac{\partial\mathcal{L}_{\text{NLL}}}{\partial\mu} = \frac{\mu-y}{\sigma^2} \tag{34}$$

Under NLL training, the gradient magnitude is scaled by $1/\sigma^2$, allowing the model to modulate learning signals through the variance prediction.

Theoretical Analysis of Lemma 5.1.

Lemma A.9 (**Gradient Buffering**, Restated)**.** *Under NLL training, the effective gradient on μ scales as $O(1/\sigma^2)$. For users with narrow score distributions where training signals provide weak discriminative information, the model learns to increase σ^2, thereby attenuating gradient magnitudes and preventing the collapse that occurs in deterministic models trained with MSE.*

Proof. Consider a user u whose ground-truth reward distribution has low variance (narrow distribution with small $\Delta_y := y_{\max} - y_{\min}$).

Under MSE training, when Δ_y is small and inputs x_i are diverse, the model faces conflicting gradients: different inputs with similar labels pull μ toward the same value. The equilibrium is a near-constant prediction $\mu_{\text{MSE}}(x) \approx \bar{y}$, constituting user-level collapse.

Under NLL training, the model jointly optimizes μ and σ^2. The gradient with respect to σ^2 is:

$$\frac{\partial \mathcal{L}_{\text{NLL}}}{\partial \sigma^2} = \frac{1}{2\sigma^2} - \frac{(y-\mu)^2}{2\sigma^4} \tag{35}$$

Setting this to zero yields the per-sample equilibrium $\sigma^{*2} = (y-\mu)^2$. When the model cannot accurately predict μ, the residual $(y-\mu)^2$ remains large, and σ^2 increases to match this residual. The gradient on μ then becomes:

$$\frac{\partial \mathcal{L}_{\text{NLL}}}{\partial \mu} = \frac{\mu - y}{\sigma^2} = O(1/|y-\mu|) \tag{36}$$

As σ^2 increases, gradient magnitudes on μ decrease, preventing the model from being forced to fit contradictory signals and allowing μ to maintain meaningful variation. □

Theoretical Analysis of Lemma 5.2.

Lemma A.10 (**Implicit Regularization**, Restated)**.** *Under NLL training, the learned variance σ^2 acts as an adaptive sample weight, effectively down-weighting samples with unreliable or inconsistent labels. This implicit regularization prevents overfitting to spurious patterns, resulting in mean predictions μ that exhibit higher correlation with true rewards and reduced reward hacking.*

Justification. The NLL loss can be decomposed as:

$$\mathcal{L}_{\text{NLL}} = \frac{1}{2}\log\sigma^2 + \frac{1}{2\sigma^2}(y-\mu)^2 \tag{37}$$

The second term is a weighted MSE with sample-dependent weight $w = 1/\sigma^2$. For samples where the input-output relationship is inconsistent or noisy, the model cannot achieve low residuals through μ alone. The optimal response is to increase σ^2 for these samples, which reduces their contribution to the MSE term while incurring a modest penalty from the $\log\sigma^2$ term (preventing $\sigma^2 \to \infty$).

Spurious patterns arise when the model memorizes incidental correlations that do not generalize. Under NLL training, precisely these noisy samples receive high σ^2 predictions, reducing their influence on μ. This prevents the model from confidently learning spurious correlations. By down-weighting noisy samples during training, the resulting μ predictions are more aligned with genuine quality signals, yielding higher correlation ρ_q and lower hacking rate β. □

Remark. A key insight is that the variance head σ^2 functions as an auxiliary task that improves learning of μ without directly participating in inference-time decisions. Both gradient buffering and implicit regularization operate entirely during training. At inference time, we select candidates using only μ, but this μ is more robust than one trained with MSE alone. Notably, using σ^2 at inference time (e.g., filtering out high-variance predictions) does not further improve performance in our experiments, reinforcing that the primary value of probabilistic modeling lies in training-time regularization.

B Additional Experimental Results

B.1 Generalization Across Policy Families

The main-text results use a RAG policy. To probe whether TTP gains depend on the choice of generator, we additionally evaluate two other policy families: *Persona prompting* (a user-style description synthesised from 15 history samples by an external LLM) and *Persona+RAG* (concatenating both). Figure A1 shows the corresponding scaling curves; Tables A2 and A3 list the numerical Best-of-N=30 results.

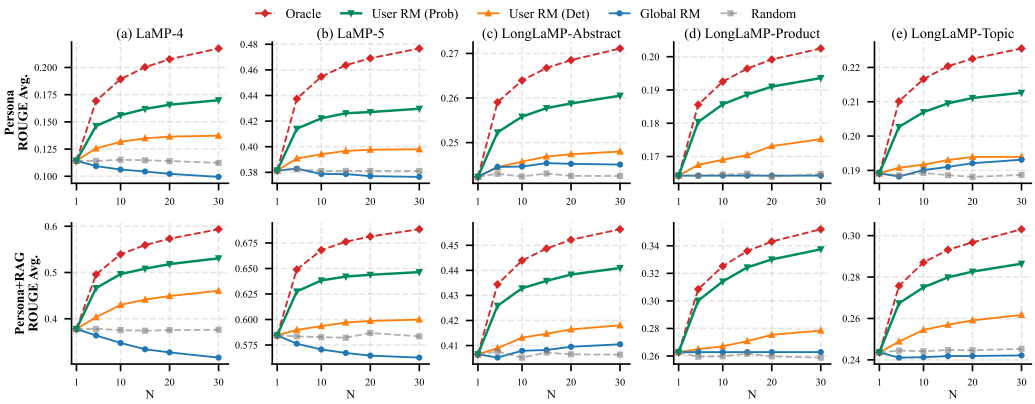

Figure A1: TTP scaling under two additional policy families across all five tasks. Top row: Persona prompting (weak N=1 baseline). Bottom row: Persona+RAG (strong N=1 baseline). The selector ranking Probabilistic $>$ Deterministic $>$ Global $\approx$ Random is preserved on every (task, policy) pair, confirming that TTP acts as a policy-orthogonal amplifier whose effect is determined by the reward model rather than the underlying generator.

"Capture" below denotes (Prob $-$ Random)/(Oracle $-$ Random), the fraction of the oracle gap recovered by the probabilistic reward model.

Table A2: Persona prompting policy: ROUGE at N=30.

Task	Random	Global	Det.	**Prob.**	Oracle	Capture
LaMP-4	0.112	0.099	0.137	**0.170**	0.218	55%
LaMP-5	0.381	0.376	0.398	**0.430**	0.477	51%
Abstract	0.243	0.245	0.248	**0.261**	0.271	63%
Topic	0.189	0.193	0.194	**0.213**	0.226	65%
Product	0.165	0.164	0.175	**0.194**	0.203	76%

Table A3: Persona+RAG policy: ROUGE at N=30.

Task	Random	Global	Det.	**Prob.**	Oracle	Capture
LaMP-4	0.376	0.316	0.460	**0.531**	0.594	71%
LaMP-5	0.584	0.563	0.600	**0.647**	0.689	60%
Abstract	0.406	0.411	0.418	**0.441**	0.456	69%
Topic	0.245	0.242	0.262	**0.286**	0.303	71%
Product	0.259	0.263	0.278	**0.337**	0.352	84%

The probabilistic User RM is the best non-oracle selector on every (task, policy) pair. Oracle capture exceeds 50% in all 10 cells and reaches 84% on Persona+RAG Product Review. The capture rate is highest for the strongest policy (Persona+RAG), confirming that stronger generators provide more high-quality candidates the reward model can exploit. Crucially, even on Persona+RAG with N=1 baselines as high as 0.376, Global RM still hacks (LaMP-4: 0.376 $\rightarrow$ 0.316), confirming that hacking is a property of the reward model rather than of the candidate quality range.

TTP with a per-user LoRA training-based policy. We further evaluate TTP on top of a strong training-based personalization method: per-user LoRA fine-tuning of the policy [4], which substantially raises the $N{=}1$ baseline. Figure A2 compares RAG-based and LoRA-based generation on LaMP-4 under the same probabilistic User RM. The LoRA policy reaches a higher oracle ceiling but its $N{=}1$ baseline is also much higher; the probabilistic User RM continues to scale, with reduced magnitude reflecting distribution shift between the RAG-trained RM and LoRA-generated candidates. This is consistent with Theorem 3.1, which bounds TTP gain by $\bar{\sigma}\sqrt{\ln N}$: a stronger policy reduces $\bar{\sigma}$ and therefore the achievable headroom, but does not invalidate the scaling mechanism.

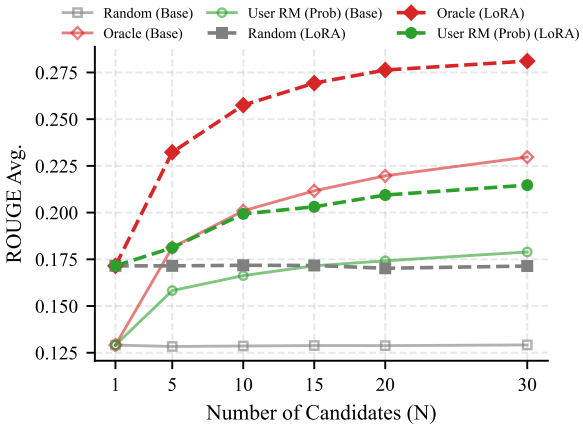

Figure A2: TTP scaling curves comparing RAG-based (Base, dashed) and LoRA-based (solid) policy models on LaMP-4. The LoRA policy reaches a higher oracle ceiling but its baseline is also much higher; the probabilistic User RM continues to scale, with reduced magnitude reflecting distribution shift between the RAG-trained RM and LoRA-generated candidates.

B.2 Robustness to Evaluation Metric

ROUGE provides a convenient ground-truth proxy but may not fully capture semantic or stylistic preferences. We replicate the central findings under two alternative metrics.

BERTScore correlates with ROUGE and replicates the trend. We compute BERTScore (`all-MiniLM-L6-v2` embeddings, cosine similarity) for all generated candidates and recompute the per-task Pearson correlation with ROUGE. Table A4 shows moderate-to-high correlation across all five tasks. On LaMP-4 at $N{=}30$, the BERTScore-rated outputs of selectors are: Random 0.372, Global 0.348 (-5.7%), Deterministic 0.362 (-2.1%), **Probabilistic** 0.392 ($+6.0\%$), ROUGE-Oracle 0.410. The ranking Prob $>$ Random $>$ Det $>$ Global is preserved, and Global RM's negative scaling under BERTScore independently confirms reward hacking is not a ROUGE artefact.

Table A4: BERTScore–ROUGE correlation on candidates pooled across all users and queries.

Task	Pearson	Spearman
LaMP-4	0.636	0.611
LaMP-5	0.664	0.656
Abstract	0.568	0.540
Topic	0.415	0.406
Product	0.448	0.447

LLM-as-judge fails for stylistic personalization. We additionally evaluated a strong general-purpose LLM (GPT-4o-mini, given user-history context) as a pointwise scoring judge (1–10 scale) on LaMP-4. The Pearson correlation between judge scores and ROUGE is $r = 0.099$, essentially random. The ground-truth answer ranks at average position $12.7/30$ (random would be 15.5) and lands in the top-5 only 34.7% of the time. Under Best-of-$N{=}30$ judge selection, ROUGE is 0.144, compared to Random 0.134 and Probabilistic 0.179. General-purpose LLMs cannot capture user-level stylistic preferences from limited history, motivating per-user trained reward models.

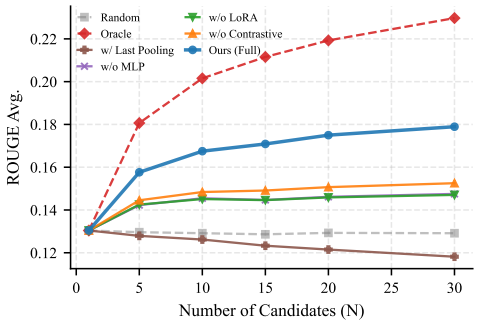

Figure A3: Ablation study: TTS performance vs number of candidates N on LaMP-4. The full model (green) consistently outperforms all ablated variants. Last-token pooling (purple) performs worse than random selection, demonstrating catastrophic failure when holistic sequence understanding is lost.

B.3 Ablation Studies

Architectural components. We conduct ablation studies on LaMP-4 to understand the contribution of each component in our probabilistic reward model. Figure A3 illustrates how each variant scales with the number of candidates N. The full model consistently outperforms all ablated variants across all values of N, with the performance gap widening as N increases. Notably, the last-pooling variant shows *negative* scaling—its performance decreases with larger N—indicating that it learns a reward function that is inversely correlated with true preferences for many queries, a manifestation of the reward hacking phenomenon discussed in Section 4.

Pooling Strategy is Critical. The most striking finding is that replacing mean pooling with last-token pooling causes catastrophic failure, with performance dropping *below* the random baseline (12.46% vs 13.58%). This demonstrates that for reward modeling in personalization tasks, aggregating information across all tokens is essential. Last-token pooling, commonly used in decoder-only language models, fails to capture the holistic semantic relationship between query and candidate, which is crucial for accurate preference prediction.

LoRA Adaptation Enables Personalization. Removing the LoRA adapter results in a substantial 3.43% drop in ROUGE-1, reducing oracle capture from 49.1% to 17.3%. This confirms that adapting the backbone representations to individual users is necessary for learning personalized preferences. Without LoRA, the model relies solely on the pre-trained representations, which cannot distinguish between different users' preference patterns.

MLP Head Improves Expressiveness. Replacing the MLP head with a simpler architecture causes a similar 3.36% performance drop. The two-layer MLP with GELU activation provides the necessary nonlinearity to map pooled representations to accurate reward predictions. This is particularly important for modeling the complex, user-specific scoring functions required for personalization.

Contrastive Learning Provides Additional Gains. Removing the high-score contrastive loss results in a 2.83% drop, the smallest among all ablations. While contrastive learning helps the model better distinguish between high-quality candidates, the primary gains come from the architectural choices (pooling, LoRA, MLP) rather than the auxiliary loss. Nevertheless, the contrastive objective provides meaningful improvements by encouraging the model to learn finer-grained preference distinctions.

Loss decomposition: NLL vs. contrastive. The architectural ablations in Appendix B.3 fix the loss as NLL+contrastive and vary the architecture. Here we hold the architecture fixed and vary the loss to decompose the contributions of NLL and the contrastive term. Results on LaMP-4 are reported in Table A5.

Two findings emerge. First, NLL is the primary contributor: replacing MSE with NLL alone closes most of the gap from 0.139 to 0.156 (Pearson lifts from 0.790 to 0.916), consistent with the prediction that NLL training prevents collapse via gradient buffering (Lemma 5.1). Second, contrastive loss is complementary but only effective on top of NLL: the same contrastive term added to MSE produces no improvement, while added to NLL it lifts ROUGE from 0.156 to 0.179. NLL

Table A5: Loss-decomposition ablation on LaMP-4 (Probabilistic User RM architecture, r=8).

Loss	RM Pearson	ROUGE@N=30	Gain over Random
MSE only	0.790	0.139	+8.2%
MSE + Contrastive	0.790	0.139	+8.0%
NLL only	0.916	0.156	+21.4%
NLL + Contrastive (full)	**0.916**	**0.179**	**+39.3%**

provides the variance-aware foundation; contrastive refines ranking in the high-quality region critical for Best-of-N.

B.4 General LLMs as Personalized Reward Models

To probe whether a strong general-purpose LLM with explicit user-persona context can replace per-user trained reward models, we evaluated a SynthesizeMe-style pairwise judge on LaMP-4: the judge LLM (`Qwen3-4B` with user-persona prompt) compares two candidates at a time, and a single-elimination knockout tournament selects the best of N=30 candidates per query (29 LLM calls per query).

Table A6: Persona-conditioned LLM tournament vs. trained reward models on LaMP-4 (Best-of-N=30).

Method	Pairwise acc.	ROUGE@N=30	Gain	Cost/query
Random	0.500	0.155	—	—
LLM tournament (Qwen3-4B)	**0.499**	0.157	+1.3%	1.63 s
LLM pointwise (GPT-4o-mini)	—	0.144	−7.1%	30 calls
Probabilistic User RM	—	**0.179**	**+15.5%**	**0.05** s

The LLM tournament's pairwise accuracy against ROUGE-determined preferences is 0.499, indistinguishable from chance. The corresponding ROUGE gain is negligible (+1.3%). The pointwise GPT-4o-mini judge performs even worse, with Pearson $r = 0.099$ to ROUGE (Appendix B.2). Both findings indicate that general-purpose LLMs cannot capture per-user stylistic preferences from limited history. The trained probabilistic reward model is also 33× faster per query. Our diagnostic framework predicts these failures: with $\rho \approx 0$ between LLM scores and true rewards, Proposition 4.4 yields flat scaling regardless of N.

B.5 Sensitivity Analyses and Statistical Stability

Sensitivity to LoRA rank. A natural question is whether reducing LoRA rank could mitigate Deterministic User RM's collapse and hacking by limiting overfitting capacity. Table A7 sweeps the LoRA rank $r \in \{2, 4, 8, 16\}$ for Deterministic User RM on LaMP-4 and contrasts with Probabilistic User RM at the same r=8 baseline.

Table A7: LoRA rank sweep on Deterministic User RM (LaMP-4).

Variant	RM Pearson	ROUGE@N=30	Gain over Random
Det. r=2	0.733	0.136	+6.0%
Det. r=4	0.735	0.138	+7.4%
Det. r=8	0.790	0.139	+8.2%
Det. r=16	0.758	0.138	+7.1%
Prob. r=8	**0.916**	**0.179**	**+39.3%**

Reducing rank does *not* fix the underlying problem: even at r=2 (four-fold fewer parameters than r=8), Deterministic RM achieves only +6.0% over random, with $\alpha = 0.20$ and $\beta = 0.32$ unchanged. At the same r=8, Probabilistic RM achieves +39.3%. The result indicates that the limitation of Deterministic User RM is not a capacity-driven overfitting issue but a property of the training dynamics under MSE loss with low-variance labels (Lemma 5.1).

Sensitivity to per-user data volume. We test whether TTP requires extensive per-user data by downsampling each user's reward-model training set on LaMP-4 to $\{10\%, 25\%, 50\%, 100\%\}$, leaving the validation split unchanged. Table A8 reports the resulting Pearson correlation of the trained Probabilistic User RM against ROUGE and the Best-of-N ROUGE at several values of N.

Table A8: Probabilistic User RM trained on a fraction of each user's history, evaluated on a fixed LaMP-4 validation set. The Random baseline is approximately 0.128 across all N.

Data %	RM Pearson	N=5	N=10	N=15	N=30
10%	0.671	0.134	0.133	0.133	0.130
25%	0.781	0.134	0.137	0.137	0.136
50%	0.887	0.138	0.143	0.145	0.141
100%	0.916	—	—	—	**0.179**

TTP delivers gains over Random even at 10% data ($+3.9\%$ at N=15), with super-linear scaling in data volume: doubling from 25% to 50% nearly doubles the ROUGE gain over Random ($7.0\% \rightarrow 13.4\%$ at N=15). This pattern is consistent with our predictive law: more training data raises $\bar{\rho}_+$ and reduces α, both of which lift ρ_{eff} and hence the scaling slope.

Statistical stability across repetitions. The main results of Section 6.2 are averages over 10 independent repetitions, each sampling N candidates from a pre-generated pool of size 40. Figure A4 and Table A9 report mean±std at representative N on LaMP-4 (RAG policy). All standard deviations are below 0.007, far smaller than the inter-method differences; pairwise differences between Probabilistic and the other methods are significant at $p < 0.01$ across all $N \geq 5$. The selector ranking is consistent across all repetitions and all N, confirming the stability of the conclusions reported in the main text.

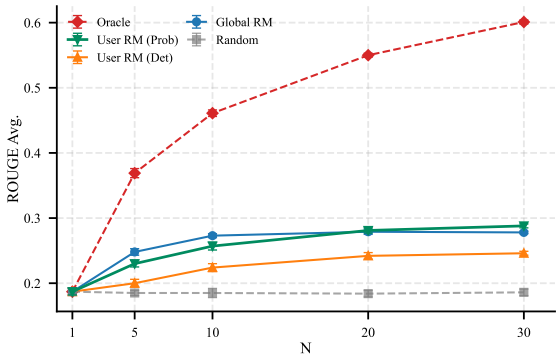

Figure A4: Best-of-N ROUGE on LaMP-4 (RAG policy) with 1σ error bars over 10 repetitions. Error bars are smaller than the marker for most points, indicating tight stability across runs.

Table A9: Mean±std over 10 repetitions on LaMP-4 (RAG policy, candidate pool size 40). Absolute values use a different pool size from Section 6.2; the relative ranking and scaling pattern are identical.

N	Random	Global	Det. User	**Prob. User**	Oracle
1	0.187 ± 0.005	0.187 ± 0.005	0.187 ± 0.005	0.187 ± 0.005	0.187 ± 0.005
5	0.185 ± 0.004	0.248 ± 0.005	0.200 ± 0.006	0.230 ± 0.005	0.369 ± 0.007
10	0.185 ± 0.007	0.273 ± 0.003	0.224 ± 0.006	0.257 ± 0.006	0.461 ± 0.005
20	0.184 ± 0.005	0.279 ± 0.004	0.242 ± 0.005	0.281 ± 0.004	0.550 ± 0.003
30	0.186 ± 0.005	0.278 ± 0.002	0.246 ± 0.003	$\mathbf{0.288 \pm 0.003}$	0.601 ± 0.003

B.6 Variance-based Selection Strategies

A natural question is whether the predicted variance from probabilistic User RM can improve candidate selection at inference time. We investigate several variance-based selection strategies and find that none outperform simple mean-based selection.

Motivation. The probabilistic User RM predicts both mean $\mu(x, y)$ and variance $\sigma^2(x, y)$ for each candidate. Intuitively, the variance captures the model's uncertainty about its prediction. One might expect that incorporating uncertainty could lead to better selection decisions, similar to exploration-exploitation trade-offs in bandit problems.

Strategies Evaluated. We evaluate the following variance-based selection strategies:

- **Mean Only**: Select $\arg\max_y \mu(x, y)$ (baseline).
- **LCB-β** (Lower Confidence Bound): Select $\arg\max_y[\mu(x, y) - \beta \cdot \sigma(x, y)]$, preferring candidates with high mean and low uncertainty.
- **UCB-β** (Upper Confidence Bound): Select $\arg\max_y[\mu(x, y) + \beta \cdot \sigma(x, y)]$, allowing exploration of uncertain candidates.
- **Var-Filter-p**: Filter out candidates in the top $p\%$ variance, then select by mean among remaining candidates.
- **SNR** (Signal-to-Noise Ratio): Select $\arg\max_y[\mu(x, y)/\sigma(x, y)]$.

Results. Table A10 shows the results on LaMP-4. All variance-based strategies perform worse than or equal to mean-only selection.

Table A10: Comparison of variance-based selection strategies on LaMP-4. All strategies underperform simple mean-based selection, indicating that variance does not provide useful signal for candidate selection at inference time.

Strategy	N=1	N=5	N=10	N=15	N=20	N=30
Oracle	.129	.181	.201	.212	.219	.230
Mean Only	.129	**.158**	**.166**	**.172**	**.174**	**.179**
LCB-0.5	.129	**.158**	**.166**	.171	.173	.178
LCB-1.0	.129	.156	.165	.169	.171	.175
LCB-2.0	.129	.154	.161	.164	.166	.169
UCB-0.5	.129	**.158**	**.166**	**.172**	**.174**	**.179**
UCB-1.0	.129	.157	.165	.170	.172	.178
UCB-2.0	.129	.155	.162	.167	.169	.174
Var-Filter-10%	.129	.152	.163	.166	.169	.173
Var-Filter-20%	.129	.152	.160	.164	.167	.171
Var-Filter-30%	.129	.146	.157	.160	.164	.168
SNR	.129	.150	.155	.156	.156	.157

Discussion. The failure of variance-based strategies suggests that while variance prediction is beneficial during training (as part of the NLL loss), the learned variance lacks proper calibration for use at inference time. Specifically:

- **LCB strategies** assume that low variance indicates reliable predictions, but this relationship may not hold if variance is not well-calibrated.
- **UCB strategies** assume high variance candidates have potential upside, but without proper calibration, high variance may simply indicate poor predictions.
- **Variance filtering** removes potentially good candidates that happen to have high predicted variance.
- **SNR** is particularly sensitive to variance scale, performing worst among all strategies.

As our probabilistic User RM is not trained with variance-aware ranking objectives, there is no guarantee that predicted uncertainty aligns with selection quality. For high-quality candidates (i.e., those with large $\mu(x, y)$), the variance is often already low or relatively uniform. In this regime, ranking is dominated by differences in the mean, leaving little room for variance-aware strategies to meaningfully reorder candidates. This may explain why LCB/UCB with small β values closely match mean-only selection, while larger β values increasingly hurt performance.

B.7 Computational Efficiency Analysis

We measure the computational costs of training-based personalization and TTP to understand their efficiency characteristics.

Experimental Setup. We benchmark both approaches on LaMP-4 using an NVIDIA H100 80GB GPU. For training-based personalization, we fine-tune per-user Qwen3-4B policy models with LoRA adapters (rank 16, sequence length 256). For TTP, we train per-user probabilistic reward models based on Qwen2.5-1.5B (sequence length 128). We measure per-sample training time and per-query inference time across 10 users. Table A11 reports the measured unit costs.

Table A11: Measured computational costs for training and inference.

Component	Model	Unit Cost
Policy Training	Qwen3-4B	59 ms/sample
RM Training	Qwen2.5-1.5B	19 ms/sample
Policy Inference (generation)	Qwen3-4B	1498 ms/query
RM Inference (scoring)	Qwen2.5-1.5B	2.3 ms/candidate

Key Findings. Our measurements reveal two important characteristics:

1. **Training efficiency**: The reward model trains $3.1\times$ faster per sample than the policy model (19 ms vs. 59 ms), owing to the smaller backbone (1.5B vs. 4B parameters) and shorter sequence length.
2. **Inference cost structure**: The RM scoring cost is negligible compared to generation cost. Scoring a single candidate takes only 2.3 ms, while generating one response takes 1498 ms—a $650\times$ difference. Even with $N = 30$ candidates, the total scoring time (69 ms) remains less than 5% of a single generation.

These findings suggest that the computational bottleneck of TTP lies entirely in candidate generation, not in reward model scoring. In deployment scenarios where candidates can be pre-generated, cached, or shared across users, TTP's inference overhead reduces to just the scoring component, making it highly efficient for high-throughput personalization.

C Theoretical Validation

C.1 Quantitative Validation of the Predictive Scaling Law

We provide the per-task evidence supporting the aggregate result reported in Section 6.2. For each task, we calibrate the scale parameter $c\bar{\sigma}$ once from the Oracle scaling curve, measure $(\alpha, \beta, \bar{\rho}_+, \bar{\rho}_-)$ from each reward model on the validation split, and substitute the values into Proposition 4.4 to predict the full Best-of-N curve. Table A12 reports the four diagnostic quantities, the effective correlation $\rho_{\text{eff}} = (1-\beta)\bar{\rho}_+ - \beta|\bar{\rho}_-|$, and the relative mean absolute error (relMAE) between predicted and observed ROUGE for $N \in \{1, 5, 10, 15, 20, 30\}$.

Across all 15 (task, RM) configurations, the formula achieves average relMAE of 2.9% with $R^2 > 0.9$ in 13 of 15 cases. The two cases with $R^2 < 0.9$ are LaMP-4 Global RM and LaMP-5 Global RM, where extreme reward hacking ($\rho_{\text{eff}} \approx 0$) yields nearly flat curves whose remaining variance is dominated by sampling noise rather than systematic scaling.

C.2 Validating the Theoretical Assumptions

Our derivations rely on two analytical assumptions: (i) the per-user true-reward distribution is sub-Gaussian (used in Theorem 3.1 and inherited by Lemma 4.2), and (ii) the joint distribution of predicted and true rewards has approximately linear conditional expectation (used in the Bivariate-Gaussian-style derivation of Lemma 4.2). Below we empirically verify that both assumptions hold to a useful degree on our data.

Table A12: Per-task validation of Proposition 4.4. The four diagnostic quantities are measured directly from each reward model; relMAE compares predicted vs. observed Best-of-N ROUGE.

Task	**RM**	α	β	$\bar{\rho}_+$	$\bar{\rho}_-$	ρ_{eff}	**relMAE**
LaMP-4	Global	0.33	0.45	0.222	−0.214	0.017	8.8%
	Det.	0.20	0.33	0.273	−0.202	0.093	1.4%
	Prob.	0.00	0.08	0.523	−0.168	0.467	2.8%
LaMP-5	Global	0.17	0.46	0.300	−0.261	0.035	7.4%
	Det.	0.07	0.37	0.312	−0.266	0.090	1.8%
	Prob.	0.00	0.22	0.458	−0.242	0.307	0.6%
Abstract	Global	0.00	0.37	0.323	−0.254	0.110	0.4%
	Det.	0.00	0.17	0.496	−0.295	0.363	1.6%
	Prob.	0.00	0.01	0.758	−0.187	0.754	0.6%
Topic	Global	0.05	0.24	0.634	−0.358	0.380	4.0%
	Det.	0.25	0.20	0.453	−0.304	0.224	1.6%
	Prob.	0.00	0.01	0.787	−0.223	0.772	1.1%
Product	Global	0.00	0.04	0.486	−0.122	0.462	1.3%
	Det.	0.25	0.19	0.324	−0.254	0.162	6.9%
	Prob.	0.00	0.00	0.757	−0.213	0.757	3.4%
Avg.	Global	0.11	0.31	0.39	−0.24	0.20	**4.4%**
	Det.	0.15	0.25	0.37	−0.26	0.19	**2.7%**
	Prob.	**0.00**	**0.06**	**0.66**	**−0.21**	**0.61**	**1.7%**

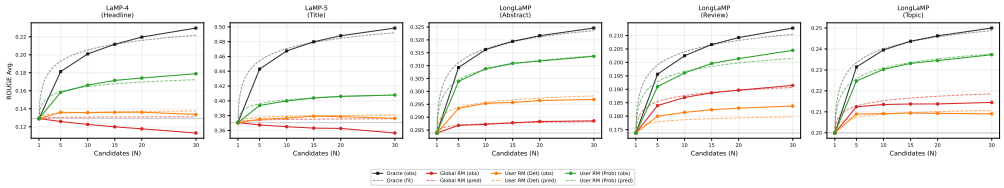

Figure A5: Predicted (dashed) vs. observed (solid) Best-of-N ROUGE for all 15 (task, RM) configurations. Predictions use the four diagnostic quantities from Table A12 substituted into Proposition 4.4; the scale parameter $c\bar{\sigma}$ is calibrated once per task from Oracle and shared across all reward models.

Sub-Gaussian property of true rewards. For each user we fit the smallest sub-Gaussian parameter σ that satisfies the moment-generating-function bound and compare the resulting tail behavior against the empirical reward distribution. We illustrate this on LaMP-4 in Figure A6: the empirical CDF lies close to the Gaussian envelope (small deviations only in the extreme tails), consistent with bounded-but-light-tailed ROUGE-derived rewards; the same pattern holds on the other four tasks. Although the bound becomes loose at the very tail, this only inflates the constant in front of $\sqrt{\ln N}$ and does not change the scaling order.

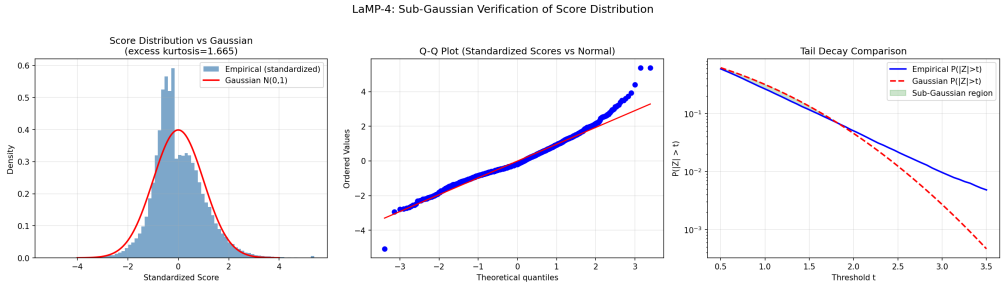

Figure A6: Sub-Gaussian property check on a representative LaMP-4 user. The empirical reward CDF (blue) is overlaid on the best-fitting Gaussian envelope (orange). Deviations are small except in the extreme tails, supporting the sub-Gaussian assumption used in Theorem 3.1.

Linear conditional expectation of true reward given predicted reward. Lemma 4.2 relies on $\mathbb{E}[r^* \mid \hat{r}]$ being approximately linear in $\hat{r}$ (this is exact under joint Normality and weakly required under any second-order linear regression structure). We illustrate this on LaMP-4 in Figure A7, binning predictions $\hat{r}$ and plotting the conditional mean of r^* against the bin centre for each of the three reward-model variants. The binned conditional means lie close to a straight line whose slope matches the empirical correlation ρ used in our analysis. Departures from linearity are small and concentrated at the rare extremes, consistent with the remark in Appendix A.2 that high-score regions slightly under-perform the global correlation.

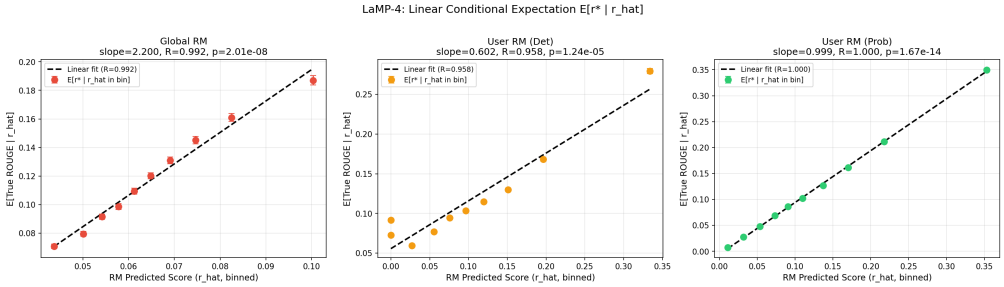

Figure A7: Bivariate linearity check on LaMP-4: empirical $\mathbb{E}[r^* \mid \hat{r}]$ vs. predicted reward $\hat{r}$, computed by binning the predictions of each reward model (Global, Det, Prob). The binned conditional means track a straight line whose slope matches the empirical correlation ρ, supporting the linear-conditional-expectation assumption in Lemma 4.2.

Together, these checks ensure that the formal derivations in Appendices A.1 and A.2 are not artefacts of overly idealised assumptions.

C.3 Detailed Failure-Mode Analysis

Cross-task user-level collapse. Figure A8 extends our user-level collapse analysis to four additional tasks. The results consistently support our theoretical findings across diverse personalization scenarios.

Collapse Identification. Across all tasks, we observe a clear separation in score standard deviation (σ) between collapsed and normal users. LongLaMP-Product Review (c) exhibits the most pronounced separation, with collapsed users showing extremely low variance ($\sigma \approx 0.13$) compared to normal users ($\sigma \approx 0.17$). This aligns with the task's nature where deterministic models tend to predict similar review scores regardless of product differences.

Deterministic RM Failure on Collapsed Users. The middle row demonstrates that deterministic RMs consistently fail on collapsed users, producing near-degenerate predictions concentrated at a single value. This is most severe in LongLaMP-Product Review (c), where the deterministic RM achieves a *negative* correlation ($\rho = -0.31$) on collapsed users, indicating predictions that are worse than random. In contrast, the probabilistic RM maintains strong correlations even for collapsed users: $\rho = 0.83$ (LaMP-5), $\rho = 0.96$ (Abstract), $\rho = 0.98$ (Product Review), and $\rho = 0.89$ (Topic Writing).

Robust Performance on Normal Users. For normal users (bottom row), both RMs perform reasonably well, though the probabilistic RM consistently achieves higher correlations. Notably, LongLaMP-Abstract (b) shows the smallest gap between methods on normal users ($\rho = 0.75$ vs 0.95), suggesting that academic writing styles may be more predictable, reducing the advantage of uncertainty modeling for well-behaved users.

Cross-task query-level reward hacking. Figure A9 presents query-level analysis across four additional tasks, validating the generality of our uncertainty-aware selection mechanism.

Correlation Between Uncertainty and Error. The second row shows strong positive correlations between learned variance σ^2 and prediction error across all tasks. LongLaMP-Product Review (c) achieves the highest correlation ($r = 0.81$), indicating that the probabilistic RM learns particularly well-calibrated uncertainty estimates for this task. The other tasks show moderate correlations ($r = 0.45$–0.51), confirming that the model successfully learns to express higher uncertainty for difficult queries.

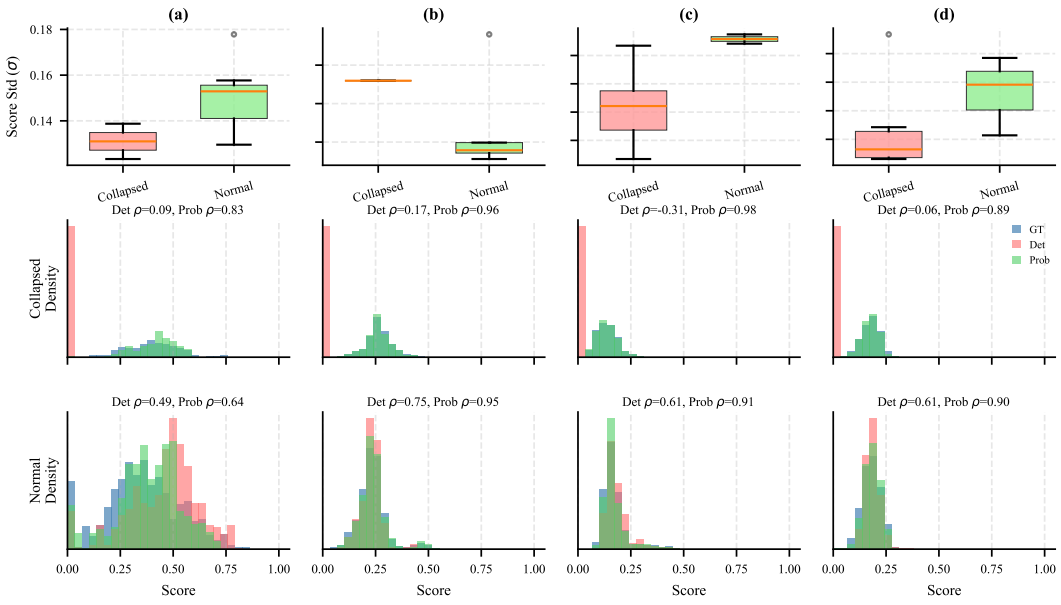

Figure A8: User-level collapse analysis across four tasks: (a) LaMP-5, (b) LongLaMP-Abstract, (c) LongLaMP-Product Review, (d) LongLaMP-Topic Writing. Top row: score standard deviation separating collapsed vs normal users. Middle/Bottom rows: score distributions for collapsed and normal users respectively, with Pearson correlations for deterministic (Det) and probabilistic (Prob) RMs.

Dramatic Reduction in Negative Correlations. The bottom two rows compare the distribution of per-query correlations. The deterministic RM exhibits substantial negative correlation rates: 38.3% (LaMP-5), 18.1% (Abstract), 28.6% (Product Review), and 26.5% (Topic Writing). These represent queries where the RM's ranking is *inversely* related to true quality, a severe failure mode for Best-of-N selection.

The probabilistic RM dramatically reduces these failure cases: negative correlations drop to 25.4% (LaMP-5), 0.4% (Abstract), 0.1% (Product Review), and 1.5% (Topic Writing). The improvement is most striking for the LongLaMP tasks, where negative correlations are nearly eliminated ($< 2\%$).

Task-Specific Observations. LaMP-5 (a) shows the smallest improvement (38.3% $\rightarrow$ 25.4%), likely because title generation from abstracts has inherent ambiguity that even uncertainty-aware methods cannot fully resolve. In contrast, LongLaMP tasks benefit more substantially, possibly because longer outputs provide richer signals for uncertainty estimation.

Rescue Analysis. The scatter plots (top row) reveal that the probabilistic RM "rescues" many queries from the hacking region (blue points above the diagonal), while rarely degrading performance (orange points below diagonal are sparse). This asymmetry confirms that uncertainty-aware selection provides a safety mechanism against reward hacking without sacrificing performance on well-predicted queries.

A priori predictability. The analyses in Appendices C.3 and C.3 document *which* users collapse and *which* queries get hacked. This subsection asks the stronger question: can these failures be predicted *a priori* from training-data statistics, before running any Best-of-N experiment?

Predicting user-level collapse. Users with low ground-truth ROUGE variance are predicted to collapse more often, since narrow label distributions provide weak discriminative signal under MSE training (Lemma 5.1). Table A13 verifies this on all five tasks: low-variance users collapse $1\times$–$13\times$ more often than high-variance users under Deterministic RM, while Probabilistic RM eliminates collapse entirely on every task regardless of label variance.

Predicting query-level hacking. Queries with low candidate-quality spread are predicted to hack more often, since limited training signal drives the model toward spurious patterns (Lemma 5.2). Table A14 verifies this prediction. Across the population of queries, Probabilistic RM reduces hacking

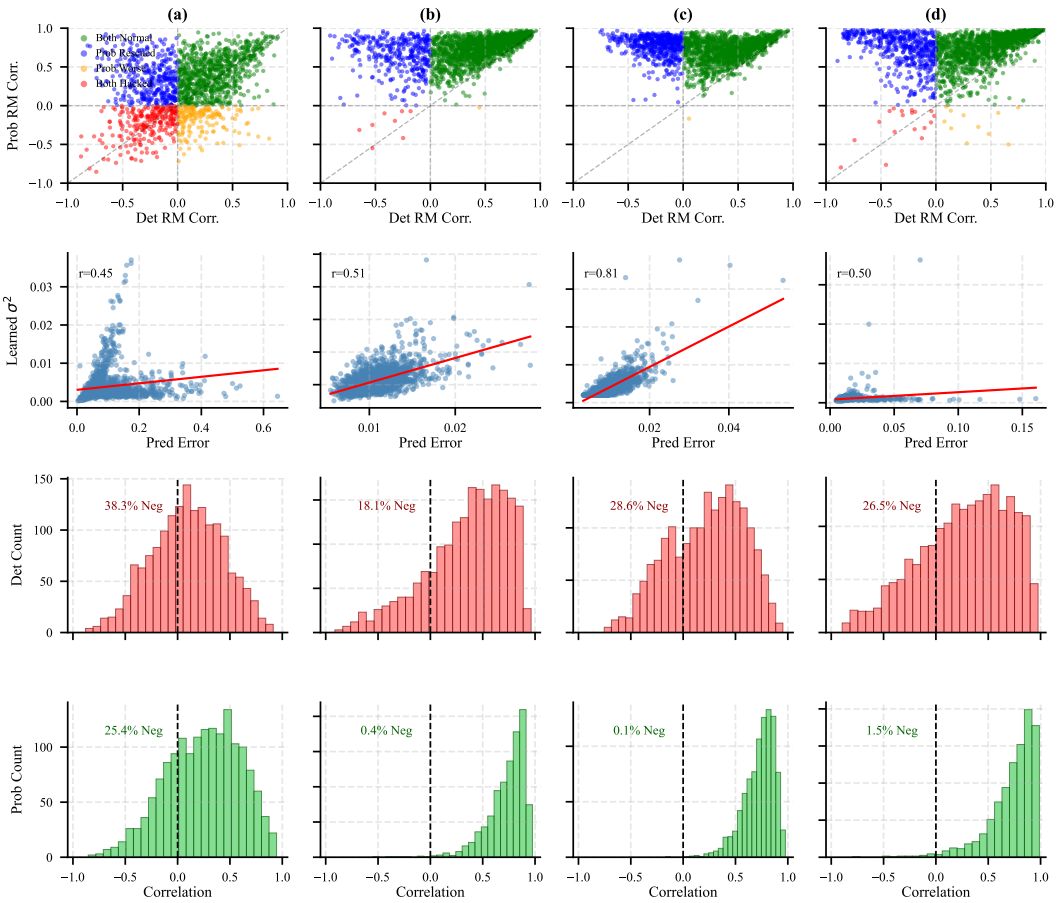

Figure A9: Query-level hacking analysis across four tasks: (a) LaMP-5, (b) LongLaMP-Abstract, (c) LongLaMP-Product Review, (d) LongLaMP-Topic Writing. Row 1: Per-query correlation comparison between Det and Prob RMs. Row 2: Learned variance vs prediction error. Rows 3–4: Distribution of per-query correlations for Det and Prob RMs, with percentage of negative correlations indicated.

Table A13: User-level collapse rates (fraction of users with $\rho_u < 0.1$), split by ground-truth ROUGE variance (low vs. high std halves). Probabilistic User RM achieves 0% collapse on every task.

	Det. User RM		**Prob. User RM**		Det. ratio
Task	Low std	High std	Low std	High std	(low/high)
LaMP-4	33.3%	6.7%	0%	0%	$5\times$
LaMP-5	13.3%	0.0%	0%	0%	$> 13\times$
Abstract	0.0%	0.0%	0%	0%	—
Topic	25.0%	25.0%	0%	0%	$1\times$
Product	30.0%	10.0%	0%	0%	$3\times$

4–$8\times$ relative to Global RM, with the largest gains precisely on the predicted high-risk queries (low candidate-quality spread).

These two tables transform the diagnostic framework into an actionable tool: practitioners can identify high-risk users and queries by computing simple per-user and per-query label statistics on the training set, before any Best-of-N experiment is run.

Table A14: Query-level hacking rates (fraction of queries with $\rho_q < 0$), split by candidate-quality spread (low vs. high std halves). Selected (task, RM) pairs.

Task	**RM**	Low spread	High spread	Reduction
Topic	Global	43.2%	7.8%	5.5×
Product	Det.	42.2%	19.8%	2.1×
LaMP-4	Det.	34.8%	31.4%	1.1×
Abstract	Prob.	1.2%	0.0%	—
Product	Prob.	<0.1%	<0.1%	—
LaMP-4	Prob.	10.9%	5.2%	2.1×

D Extrapolation Beyond the Main Setting

This section gives the full results behind Section 6.4: candidate pools up to N=100, sparse users, and quality signals other than ROUGE.

D.1 Scaling Beyond N=30

The main experiments stop at N=30. To characterize longer-range behavior and to test the scaling law out of range, we regenerated fresh 100-candidate pools with the identical decoding configuration for LaMP-4 (30 users × 100 queries) and LongLaMP-Product (20 users × 50 queries) and evaluated all selectors for N up to 100 (Figure A10, Table A15). Because the pools are freshly sampled, absolute values differ slightly from the main-text curves. For the theory curves we calibrate $c\bar{\sigma}$ from the oracle and measure the four diagnostics using *only* $N \leq 30$, then evaluate the prediction at $N \in \{50, 75, 100\}$.

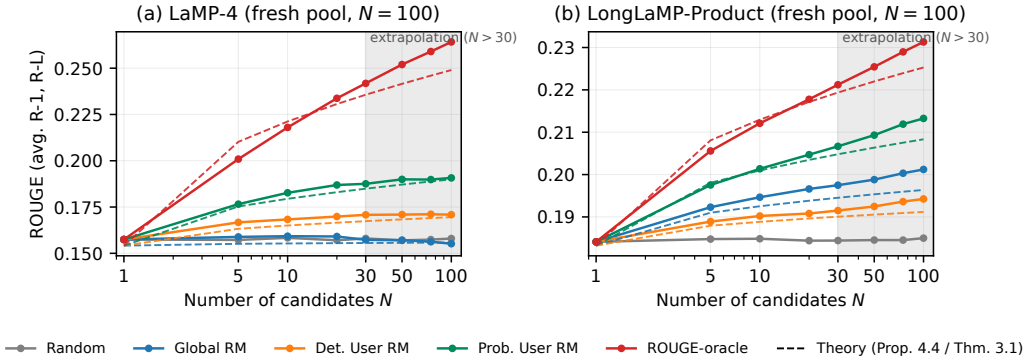

Figure A10: Best-of-N ROUGE up to N=100 on fresh candidate pools (solid: observed; dashed: Theorem 3.1 for the oracle and Proposition 4.4 for the reward models, with diagnostics measured at $N \leq 30$; shaded: extrapolation region). All three predicted regimes appear: the probabilistic User RM keeps rising, the Global RM is flat or declines below random where it hacks (LaMP-4), and the deterministic User RM saturates where it collapses (Product).

All three behaviors occur, and the diagnostics at $N \leq 30$ say which reward model shows which. On LaMP-4 the probabilistic User RM continues to improve logarithmically with no degradation; the Global RM declines below random, since with hacking rate β=0.47 a larger pool supplies more candidates on negatively-correlated queries to mis-select; the deterministic User RM saturates, the intermediate regime. LongLaMP-Product shows the complementary picture predicted by its diagnostics: the probabilistic RM rises monotonically through N=100, the Global RM scales slowly but positively (Product is the task on which it does not hack), and the deterministic RM is limited by user-level collapse (α=0.45 on the fresh pools). The oracle keeps growing as $\sqrt{\ln N}$ to N=100 with no saturation, and diagnostics measured only at $N \leq 30$ predict the N=50–100 curves at relMAE 0.5–1.2% (LaMP-4) and 1.3–2.1% (Product), an out-of-range test of Proposition 4.4.

Table A15: Observed Best-of-N ROUGE on the fresh pools, the diagnostics measured at $N \leq 30$, and the relMAE of Proposition 4.4 on the calibration range ($N \leq 30$) and on the extrapolation range ($N \in \{50, 75, 100\}$).

		ROUGE at N								Diagnostics			relMAE	
Task	**Selector**	1	5	10	20	30	50	75	100	α	β	ρ_{eff}	$N \leq 30$	$N \geq 50$
LaMP-4	Random	0.157	0.157	0.158	0.157	0.158	0.157	0.157	0.158	–	–	–	–	–
	Global	0.157	0.159	0.159	0.159	0.157	0.157	0.156	0.155	0.37	0.47	0.03	1.5%	0.5%
	Det.	0.157	0.167	0.168	0.170	0.171	0.171	0.171	0.171	0.10	0.33	0.18	1.7%	1.2%
	Prob.	0.157	0.177	0.183	0.187	0.188	0.190	0.190	**0.191**	0.00	0.16	0.38	1.3%	0.8%
	Oracle	0.157	0.201	0.218	0.234	0.242	0.252	0.259	0.264	–	–	–	–	–
Product	Random	0.184	0.185	0.185	0.184	0.184	0.185	0.185	0.185	–	–	–	–	–
	Global	0.184	0.192	0.195	0.197	0.197	0.199	0.200	0.201	0.10	0.07	0.35	1.4%	2.1%
	Det.	0.184	0.189	0.190	0.191	0.192	0.192	0.194	0.194	0.45	0.06	0.34	0.9%	1.3%
	Prob.	0.184	0.198	0.201	0.205	0.207	0.209	0.212	**0.213**	0.00	0.01	0.60	1.0%	1.9%
	Oracle	0.184	0.206	0.212	0.218	0.221	0.225	0.229	0.231	–	–	–	–	–

Operating point. Under the $\sqrt{\ln N}$ law the marginal gain per additional candidate decays as $\mathrm{d}\sqrt{\ln N}/\mathrm{d}N = 1/(2N\sqrt{\ln N})$, whereas generation cost grows linearly in N. Moving from N=30 to N=100 increases the scaling term by only $\sqrt{\ln 100}/\sqrt{\ln 30} - 1 \approx 16\%$ at 3.3× the generation cost, so N in the 5–30 range captures most of the achievable benefit; the exact choice can be made per setting from the measured diagnostics rather than by a universal cutoff.

D.2 Sparse Users

The main experiments use the users with the longest profiles (500+ items), the regime least likely to fail. Since Lemma 5.1 is about weak training signal, the sparse regime is the natural stress test. We ran the full pipeline on 30 genuinely sparse LaMP-4 users with aggregated profiles of 31–100 items, sampled evenly across that range, using leave-one-out queries (1,466 in total), identical decoding (N=30, T=1.7), identical reward-model recipes (per-user Det. and Prob. User RMs trained on each user's own candidates), and identical diagnostics. The Global RM is the main-text model applied zero-shot, since these users are unseen by it. Table A16 reports the scaling curves and Table A17 the diagnostics.

Table A16: Best-of-N ROUGE on 30 sparse LaMP-4 users (31–100 profile items).

N	Random	Global RM	Det. User RM	**Prob. User RM**	ROUGE-oracle
1	0.213	0.213	0.213	0.213	0.213
5	0.214	0.214	0.224	**0.229**	0.267
10	0.216	0.214	0.227	**0.235**	0.288
20	0.213	0.212	0.228	**0.238**	0.304
30	0.213	0.212	0.230	**0.239**	0.313

Table A17: Diagnostics on the sparse users and the resulting fit of Proposition 4.4 (relMAE over $N \in \{1, 5, 10, 20, 30\}$).

RM	α	β	$\bar{\rho}_+$	$\bar{\rho}_-$	ρ_{eff}	relMAE
Global (zero-shot)	0.43	0.49	0.31	−0.33	0.00	1.4%
Det. User RM	0.07	0.33	0.45	−0.31	0.20	0.9%
Prob. User RM	**0.03**	**0.29**	**0.49**	−0.30	**0.27**	1.5%

Three findings. **TTP remains effective on sparse users**: the probabilistic User RM gains +11.8% over random at N=30 (25% of the oracle gap; deterministic +7.7%), with the same selector ranking and monotonic scaling. The zero-shot Global RM drops to random (α=0.43, β=0.49): population-level reward signal does not transfer to unseen sparse users, which underscores the need for per-user reward models. **The reduced gain is predicted by the framework**: relative to data-rich users ($\approx 50\%$ oracle-gap capture), ρ_{eff} falls from 0.48 to 0.27 with less training signal, and substituting the sparse-measured diagnostics into Proposition 4.4 reproduces the observed curves at relMAE 0.9–1.5%. **Collapse does not spike** (α^{prob}=0.03, α^{det}=0.07), a result we did not anticipate: these users have higher label variance than the data-rich users, and by Lemma 5.1 it is low label variance, not sparsity

itself, that creates the collapse attractor. Sparsity instead manifests as elevated hacking (β=0.29–0.33 versus 0.08 for the probabilistic RM on data-rich users). The sparse regime is therefore harder, for the reason the theory identifies, and the diagnostics remain the right early-warning instrument.

D.3 Beyond ROUGE: Cross-Metric Transfer and Human Evaluation

Selection-level cross-metric transfer. Correlation between metrics (Appendix B.2) is only indirect evidence that ROUGE-based selection is meaningful. A stronger, selection-level check is to select candidates by one metric and evaluate the selected outputs with another: we select by ROUGE (the ROUGE-oracle at N=30) and score the selected outputs with BERTScore, against random selection from the same pool. If ROUGE selection merely exploited n-gram overlap, BERTScore would show no gain; instead it is positive on every task (Table A18), largest on the short-form tasks where ROUGE headroom is also largest.

Table A18: BERTScore of ROUGE-selected outputs versus random selection (N=30). Selecting by ROUGE improves the independent metric on all five tasks.

Task	Random → BERTScore	ROUGE-selected → BERTScore	Gain
LaMP-4	0.372	0.410	+10.2%
LaMP-5	0.676	0.737	+9.0%
Abstract	0.750	0.766	+2.1%
Topic	0.483	0.502	+3.9%
Product	0.582	0.601	+3.3%

Human evaluation. To test whether ROUGE-based selection aligns with human judgment, we ran a blind pairwise study. All candidates for a query are sampled from the same personalized policy, so the two responses in a pair differ *only* in the selection strategy. We built 76 pairwise items on LaMP-5 (60 items, short scholarly titles) and LongLaMP-Abstract (16 items, long-form), plus 4 catch trials (user-written reference vs. the lowest-ROUGE candidate). Each item shows the query, several of the user's own writing samples as context, and two responses in randomized A/B order; annotators pick the response that better matches the user's style and intent, with a Tie option. Test T1 (ROUGE-oracle-selected vs. randomly-selected candidate, 38 items) tests metric validity; test T2 (Prob-RM-selected vs. random, 38 items) tests the end-to-end system. Three annotators passed quality control (each $3/4$ catch trials correct; submissions failing the catch trials or an independence check were excluded and re-collected). Table A19 reports per-annotator and majority-vote win rates for the selected output, a one-sided sign test on the majority vote, and Fleiss' κ.

Table A19: Human evaluation: win rate of the selected output over a random candidate from the same pool (three annotators, 38 items per test; no annotator used the Tie option). p: one-sided sign test on the majority vote.

Test	Ann. 1	Ann. 2	Ann. 3	Majority	LaMP-5 / Abstract	p	Fleiss' κ
T1: ROUGE-oracle vs. random	63%	58%	58%	**61%** (23/38)	60% / 62%	0.13	0.38
T2: Prob. RM vs. random	53%	58%	47%	53% (20/38)	53% / 50%	0.44	0.44

Every annotator individually preferred the ROUGE-selected output in T1, on both the short-form and the long-form task, supporting directional alignment between ROUGE and human judgment; 38 items are not enough for significance, and we report the study as supporting rather than establishing alignment. T2 is near chance for a structural reason: candidates selected from the same personalized pool typically differ in only a few words, especially on LaMP-5 titles, so the differences sit near the resolution limit of human pairwise judgment even though reference-based metrics register them. Larger studies weighted toward long-form tasks, with catch trials balanced across A/B sides, are the natural next step.

E Pipeline and Implementation Details

Our experimental pipeline consists of three stages, as illustrated in Figure A11: (1) **Personalized Prompting**, where we construct personalized policy models using retrieval-augmented generation;

(2) **Reward Model Training**, where we train user-specific reward models to capture individual preferences; and (3) **Inference**, where we apply test-time personalization by sampling multiple candidates and selecting the best one via the reward model. We describe each stage in detail below.

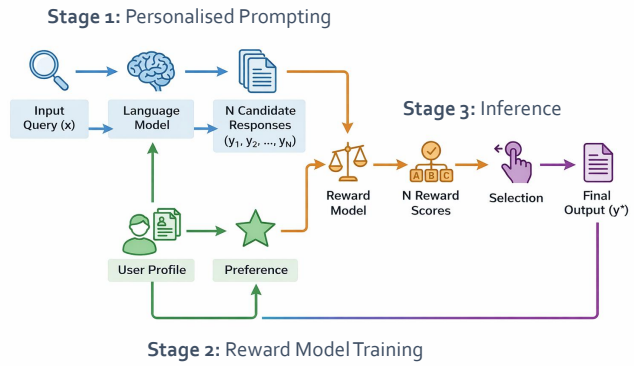

Figure A11: Overview of the experimental pipeline.

E.1 Stage 1: Personalized Prompting

The goal of personalized prompting is to generate candidate responses that serve as training data for reward models. This stage involves dataset preparation and policy model configuration.

Datasets. We evaluate on five personalized text generation tasks: **News Headline Generation** (LaMP-4) and **Scholarly Title Generation** (LaMP-5) from LaMP [3], and **Abstract Generation**, **Product Review Writing**, and **Topic Writing** from LongLaMP [13]. We exclude email-related tasks from both benchmarks as they require additional access to private email datasets. For all tasks, we adopt the *temporal-based* data split provided by the original benchmarks, which ensures that test queries are chronologically after training examples to simulate realistic deployment scenarios.

Following Tan et al. [4], we select the top-30 users with the longest profiles for LaMP tasks and top-20 users for LongLaMP tasks. Representative input-output examples for each task are shown as follows.

LaMP-4: News Headline Generation

Input: Generate a headline for the following article: According to AP, Harry's days as a pilot are over. The prince, who spent over three years training as an Apache helicopter...
Output: Prince Harry Scores A New Job, Says Goodbye To His Pilot Uniform

LaMP-5: Scholarly Title Generation

Input: Generate a title for the following abstract of a paper: We propose a novel evolutionary approach to the problem of timing-driven FPGA placement. The method used is evolutionary programming (EP) with incremental position encoded in the population...
Output: An evolutionary approach to timing driven FPGA placement

LongLaMP: Abstract Generation

Input: Generate an abstract for the title "Learned Token Pruning for Transformers" using the following items: 1. Transformer models 2. Learned Token Pruning (LTP) 3. Performance improvement 4. FLOPs Reduction 5. Robustness
Output: Efficient deployment of transformer models in practice is challenging due to their inference cost including memory footprint, latency, and power consumption, which scales quadratically with input sequence length. To address this, we present a novel token reduction method dubbed Learned Token Pruning (LTP)...

LongLaMP: Product Review Writing

Input: Generate the review text for a product with rating "5.0" and description "Kate Aster is the bestselling author of the Special Ops: Homefront series...". Summary: "Contract with a SEAL".
Output: It is impossible not to fall in love with the characters in this book, which brings in strong, supportive relationships with family and friends. Aster has done a remarkable job of making the plot and characters engaging...

LongLaMP: Topic Writing

Input: Generate the content for a reddit post titled "good riddance to bad rubbish".
Output: As an Eagle who worked very hard to encourage BSA to change it's policy, I have to say this is good news. This is symbolic of the total defeat of the Christianist/anti-gay lobby in Scouting...

Policy Model Configuration. We use `Qwen3-4B-Instruct` [15] as the policy model backbone. To achieve personalization, we adopt a retrieval-augmented generation (RAG) approach with dense retrieval [3]. Specifically, we use `all-MiniLM-L6-v2` [23] as the text encoder for embedding-based retrieval. Given an input query, we encode it and retrieve the top-k most similar examples from the user's profile based on cosine similarity. The retrieved examples are concatenated as context in the prompt.

For decoding, we use `vLLM` [24] to accelerate generation. For each query, we sample $N = 30$ candidate responses to construct reward model training data. Task-specific prompt templates are provided in Appendix F.1.

Generation Hyperparameters. Table A20 summarizes the generation hyperparameters and dataset statistics for each task.

Table A20: Dataset statistics and generation hyperparameters. Avg Len denotes the average candidate length in characters.

Task	# Users	# Queries	Cands/Query	Avg Len	Temp	Top-k	Top-p	# History
LaMP-4 (Headline)	30	24015	24.9	74	1.7	50	0.90	5
LaMP-5 (Title)	30	12036	16.8	96	1.7	50	0.90	5
LongLaMP (Abstract)	20	22388	30.0	1227	1.5	40	0.90	3
LongLaMP (Review)	20	12580	30.0	514	1.5	40	0.90	3
LongLaMP (Topic)	20	7123	29.2	1101	1.5	40	0.90	3

We set the number of candidate generations to $N = 30$ for all tasks. However, for short-form generation tasks (LaMP-4 and LaMP-5), the policy model occasionally produces duplicate outputs despite temperature sampling. We remove exact duplicates from the candidate set, resulting in an average of fewer than 30 unique candidates per query for these tasks. Long-form generation tasks (LongLaMP) exhibit minimal duplication due to their greater output diversity.

E.2 Stage 2: Reward Model Training

This stage describes the training procedure for three reward model variants: Global RM, Deterministic User RM, and Probabilistic User RM.

Model Architecture. All reward models share the same backbone architecture: `Qwen2.5-1.5B-Instruct` [16] with masked mean pooling over hidden states. For probabilistic User RM, an additional variance head predicts $\log \sigma^2$ with output passed through softplus and clamped to $[10^{-4}, 0.5]$.

Reward Label Construction. For each candidate response, we compute the ROUGE score against the ground-truth output as the reward label: $r = (\text{ROUGE-1} + \text{ROUGE-L})/2$. where ROUGE-1 measures unigram overlap and ROUGE-L measures longest common subsequence. Scores are clipped to $[0, 1]$.

Training Configurations. Table A21 summarizes the training hyperparameters for each reward model variant.

Table A21: Reward model training configurations.

Hyperparameter	Global RM	User RM (Det)	User RM (Prob)
Backbone		Qwen2.5-1.5B-Instruct	
Backbone training	LoRA	LoRA	LoRA
Learning rate (head)	10^{-4}	10^{-4}	2×10^{-4}
Learning rate (LoRA)	10^{-4}	10^{-4}	2×10^{-5}
Batch size	48	16	8
Epochs	8	10	15
Warmup ratio	0.05	0.1	0.1
Early stopping	–	–	patience=5
Weight decay	0.01	0.01	0.01
Max length (LaMP)	128	128	128
Max length (LongLaMP)	832	832	832

LoRA Configuration. For models using LoRA fine-tuning, we apply low-rank adaptation with rank $r = 8$, scaling factor $\alpha = 16$, and dropout 0.05. Target modules are `q_proj` and `v_proj`. This results in approximately 4.6M trainable parameters per user model.

Loss Functions. **Deterministic User RM** is trained with MSE loss. **Probabilistic User RM** is trained with Gaussian NLL loss (Equation 6). Two variants of user RM are also trained with a contrastive term for high-score discrimination, which the loss weighted parameter is $\lambda = 1.0$. The contrastive loss operates on sample pairs where both have ground-truth scores above 0.5, with margin $m = 0.02$. We also apply linear sample weighting with weights in $[0.3, 1.0]$ proportional to ground-truth scores, emphasizing high-quality samples. **Global RM** is trained with BCE loss with sample reweighting for extreme scores.

User Selection and Data Split. For LaMP tasks, we select the top-30 users with the most data, using up to 3,000 samples per user. For LongLaMP tasks, we select the top-20 users with up to 5,000 samples per user. All data is split 80%/20% for training/validation per user.

Implementation Details. All experiments are conducted on a single NVIDIA H100 PCIe GPU. We use BF16 mixed precision training with Flash Attention 2 [25] for efficiency. The optimizer is AdamW [26] for all models. All RMs share the same prompt templates, which are provided in Appendix F.2. No retrieval-based method is used for RM training.

E.3 Stage 3: Inference

This stage describes the Test-Time Personalization (TTP) inference procedure and evaluation protocol.

TTP Inference Procedure. Given a test query x for user u, TTP operates as follows:

1. Sample N candidates from the pre-generated candidate pool (up to 30 candidates per query).
2. Score each candidate using the user-specific reward model R_u.
3. Select the candidate with the highest predicted reward: $\hat{y} = \arg\max_{y \in \mathcal{Y}_N} R_u(x, y)$.

For probabilistic User RM, we use only the predicted mean $\mu(x, y)$ for selection.

Evaluation Protocol. Table A22 summarizes the inference configuration. For each (user, query, N) combination, we repeat the random candidate sampling 3 times and report the average to reduce variance from candidate selection.

Table A22: Inference and evaluation configuration.

Parameter	Value
N values tested	1, 5, 10, 20, 30
Queries per user	100
Random trials per N	3
Inference batch size	32

F Prompt Templates

F.1 Policy Model Templates

We adopt a retrieval-augmented generation (RAG) approach for personalized generation. For each query, we retrieve relevant examples from the user's history and construct the prompt using task-specific templates. Below we present the prompt templates for each task.

LaMP-4: News Headline Generation

```
{history}

### Instruction: Generate a headline for the following article. Match
the user's headline style.

### Article:
{input}

### Headline:
```

LaMP-5: Scholarly Title Generation

```
{history}

### Instruction: Generate a title for the following abstract. Match the
user's academic titling style. Output ONLY the title.

### Abstract:
{input}

### Title:
```

LongLaMP: Abstract Generation

```
{history}

### Instruction: Write a scientific abstract for the following title,
matching the user's academic writing style found in the history.

### Title:
{input}

### Abstract:
```

LongLaMP: Product Review Writing

```
{history}

### Instruction:
You are the user described in the [User History] above. The User Request
below specifies the product description, the rating you gave, and the
summary you wrote. Write the full Review Text that corresponds to these
details. Mimic your detailed, narrative writing style found in the
history. Write a full body review (around 50-100 words), telling the
story of your experience.

### User Request:
{input}

### Review Content:
```

LongLaMP: Topic Writing

```
{history}

### Instruction:
You are the user described in the [User History] above. Write a Reddit
post about the topic below. Mimic the user's vocabulary and tone, but
keep the response relatively short and punchy (around 100-200 words).
Do not write a wall of text.

### Topic:
{input}

### Post Content:
```

In all templates, `{history}` is replaced with retrieved user history examples, and `{input}` is replaced with the current query. The model generates text following the final prompt marker (e.g., `### Headline:`).

F.2 Reward Model Prompt

The reward model receives a structured prompt consisting of a task-specific instruction, the input query, and the candidate response:

Reward Model Prompt Template

```
[Instruction]: {task_instruction}
[Input]: {query}
[Document]: {candidate}
```

where `{task_instruction}` is one of the following task-specific instructions:

LaMP-4: News Headline Generation

```
Judge how good the Document is as a headline for the Article.  Higher score =
better.
```

LaMP-5: Scholarly Title Generation

```
Judge how good the Document is as a title for the Abstract.  Higher score =
better.
```

LongLaMP: Abstract Generation

```
Judge how good the Document is as an abstract for the Title.  Higher score =
better.
```

LongLaMP: Product Review Writing

```
Judge how good the Document is as a review for the Product.  Higher score =
better.
```

LongLaMP: Topic Writing

```
Judge how good the Document is as a post for the Topic.  Higher score =
better.
```

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The three contributions stated in the abstract and Section 6.2 are supported by Sections 3–5 (theory) and Section 6 (experiments).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See the Limitations paragraph in Section 8 (Conclusion and Discussion).

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: All theoretical results are stated with their assumptions; full proofs are in Appendix A.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Full pipeline, hyperparameters, and prompts are in Appendix E (with templates in Appendices F.1 and F.2).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Datasets are public (LaMP, LongLaMP); code and reproduction scripts will be released anonymously in the supplementary material at submission time.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Setup is summarized in Section 6; full hyperparameters are in Appendix E.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Each experiment is repeated 10 times; standard deviations and significance tests are reported in Appendix B.5.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All training uses a single NVIDIA H100 GPU; per-sample timings are reported in Appendix B.7.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research uses only public, peer-reviewed, de-identified benchmarks (LaMP, LongLaMP) and conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper does not include a dedicated discussion of societal impacts. The work studies personalized text generation on public, de-identified benchmarks (LaMP, LongLaMP) with no new data collection or model release; we view the direct societal impact as marginal beyond standard considerations for personalization technology.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release any new pre-trained models, image generators, or scraped datasets.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All existing assets (LaMP, LongLaMP, Qwen2.5/Qwen3, all-MiniLM-L6-v2, vLLM, FlashAttention-2) are cited in Section 6 and Appendix E and used under their respective public licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new public datasets, benchmarks, or model checkpoints.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: The paper adds a small human evaluation (three annotators, 80 items each) in which annotators compare two machine-generated texts for stylistic fit to a user's writing samples. Item construction, the instructions shown to annotators, catch trials, quality control, and exclusion criteria are described in the appendix; the task involved no personal or sensitive data and no deception.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The annotation task asked volunteers to judge publicly available, machine-generated text for stylistic fit; no personal data were collected and the task posed no risk beyond ordinary reading, so it did not fall under the institution's requirement for ethics review.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Pre-trained LLMs (Qwen2.5/Qwen3) are used as standard backbones; an external LLM is used only for standard prompt-based persona synthesis. No LLM is a non-standard component of the core method.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.