# SPARSE PERSONALIZED TEXT GENERATION WITH MULTI-TRAJECTORY REASONING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

As Large Language Models (LLMs) advance, personalization has become a key mechanism for tailoring outputs to individual user needs. However, most existing methods rely heavily on dense interaction histories, making them ineffective in cold-start scenarios where such data is sparse or unavailable. While external signals (e.g., content of similar users) can offer a potential remedy, leveraging them effectively remains challenging: raw context is often noisy, and existing methods struggle to reason over heterogeneous data sources. To address these issues, we introduce PAT (**P**ersonalization with **A**ligned **T**rajectories), a reasoning framework for cold-start LLM personalization. PAT first retrieves information along two complementary trajectories: writing-style cues from stylistically similar users and topic-specific context from preference-aligned users. It then employs a reinforcement learning-based, iterative dual-reasoning mechanism that enables the LLM to jointly refine and integrate these signals. Experimental results across real-world personalization benchmarks show that PAT consistently improves generation quality and alignment under sparse-data conditions, establishing a strong solution to the cold-start personalization problem.

## 1 INTRODUCTION

Large Language Model (LLM) personalization has received substantial attention due to its transformative potential in applications such as recommender systems and conversational agents (Zhang et al., 2025b; Li et al., 2024; Au et al., 2025; Kumar et al., 2024; Salemi et al., 2024). By effectively leveraging personal context, personalized LLMs can generate content tailored to individual users, enhancing the user experience and fostering deeper engagement.

Existing strategies for LLM personalization heavily rely on the assumption of rich personal histories. For example, LaMP (Salemi et al., 2024) employs a Retrieval-Augmented Generation (RAG) framework to supply historical context to the prompt, while P-RLHF (Li et al., 2024) learns user embeddings derived from extensive past interactions. However, assuming rich history availability is often not realistic in real-world settings, where thousands of new users emerge daily (Zhang et al., 2025a). Thus, a crucial challenge lies in addressing the cold-start problem—effectively personalizing for users with sparse historical data. To mitigate this data scarcity, recent work has used auxiliary information to augment the limited user context; e.g., PGraphRAG (Au et al., 2025) leverages a user-topic graph to retrieve neighbor writings related to the generation target. However, a key challenge remains: the retrieved context is often noisy and heterogeneous, encompassing signals ranging from the writing styles of similar users to distinct opinions on target topics. This complexity underscores the need for robust reasoning and sophisticated strategy to effectively integrate these heterogeneous signals into the generation process (Salemi et al., 2025).

To bridge this gap, we propose PAT, a retrieval and reasoning framework for sparse LLM personalization. Instead of treating the retrieved content as a single, monolithic context, PAT decomposes the personalization task into complementary trajectories: writing-style context from stylistically similar and topic-specific knowledge context from preference-aligned users. By employing a reinforcement-learning-based, iterative dual-reasoning mechanism, our approach enables the LLM to jointly refine and integrate these heterogeneous signals, filtering out noise while preserving critical personal markers. This allows the model to reason across multiple trajectories, making the final generation better aligned with user preferences, even under sparse conditions when the user's own history is limited.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: Aggregated results across datasets.

| | Metric | PAT (Ours) | GraSPeR | PGraph | LaMP |
|---|---|---|---|---|---|
| LONG TEXT GEN. | R-1 ↑ | **0.233** | 0.195 | 0.213 | 0.182 |
| | R-L ↑ | **0.162** | 0.147 | 0.146 | 0.124 |
| | MET ↑ | **0.187** | 0.148 | 0.171 | 0.149 |
| | LLM ↑ | **3.288** | 2.868 | 3.185 | 3.070 |
| SHORT TEXT GEN. | R-1 ↑ | **0.157** | 0.148 | 0.126 | 0.112 |
| | R-L ↑ | **0.153** | 0.145 | 0.121 | 0.106 |
| | MET ↑ | **0.128** | 0.124 | 0.124 | 0.117 |
| | LLM ↑ | **3.541** | 3.240 | 3.396 | 3.225 |

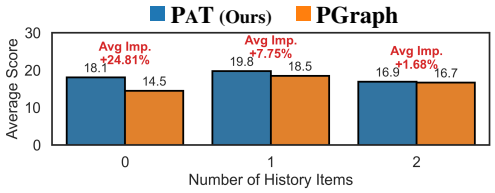



Figure 1: Performance of our approach across varying user-history sparsity levels.

As shown in Table 1 and Figure 1, experiments on three real-world datasets demonstrate that PAT outperforms state-of-the-art personalization baselines, gaining significant improvement on users with sparse history. In summary, our contributions are:

- **Framework:** We introduce PAT, a novel framework for cold-start LLM personalization that replaces simple context augmentation with multi-trajectory retrieval and reasoning.
- **Context Augmentation:** We propose a structured context augmentation strategy that decomposes retrieved heterogeneous information into complementary personalization signals.
- **Dual-Reasoning Mechanism:** We propose an iterative dual-reasoning paradigm that enables LLMs to synthesize heterogeneous style and preference signals, jointly optimizing the trajectories to improve personalization.
- **Empirical Validation:** We conduct extensive experiments on three benchmark datasets, demonstrating that our approach improves generation quality and alignment in sparse-data scenarios.

## 2 PROBLEM DEFINITION

Here, we formally define the task and establish the notation used throughout this work. Let $\mathcal{U}$ be the set of users. The full notation is provided in Appendix G. Each user $u \in \mathcal{U}$ is associated with a personal history $\mathcal{H}_u = \{(x_{u,1}, y_{u,1}), \ldots, (x_{u,n}, y_{u,n})\}$, where $x_{u,i}$ represents an input prompt and $y_{u,i}$ denotes the corresponding ground-truth text produced by the user. Additionally, let $\mathcal{A}_u$ represent any *auxiliary information* for user $u$ (e.g., similar user profiles) that can supplement the user's context.

**Definition 1** (Personalized Text Generation)**.** *Given a target user $u$, their historical context $\mathcal{H}_u$, auxiliary information $\mathcal{A}_u$, and a new target prompt $x_{target}$, the objective is to learn a parameterized function $f_\theta$ that generates a personalized output sequence $\hat{y}$:*

$$\hat{y} = f_\theta(x_{target}, \mathcal{H}_u, \mathcal{A}_u) \tag{1}$$

**Remark 1** (Cold-Start Personalization)**.** *In this work, we specifically address the cold-start scenario, where the user history $\mathcal{H}_u$ is sparse or limited in size (i.e., $|\mathcal{H}_u|$ is small). In such cases, $\mathcal{H}_u$ lacks sufficient signals for the model $f_\theta$ to effectively capture the user's specific writing style or preferences, often resulting in generic or misaligned generations. Consequently, the effective construction and utilization of auxiliary information $\mathcal{A}_u$ becomes crucial for mitigating this problem of data sparsity.*

## 3 APPROACH

Next, we introduce PAT, our proposed framework for cold-start personalization. As established in Section 2, when the local history $\mathcal{H}_u$ is sparse, the *construction* and *utilization* of auxiliary information $\mathcal{A}_u$ become the key to generation quality. Shown in Figure 2, on a high level, PAT first identifies and augments the auxiliary information $\mathcal{A}_u$ with stylistically similar and preference-aligned neighbors. To effectively leverage $\mathcal{A}_u$ for personalization, PAT then employs a multi-trajectory reasoning mechanism that enables the LLM to iteratively refine and integrate these heterogeneous signals into a personalized output. In the rest of this section, we will introduce them in more detail.

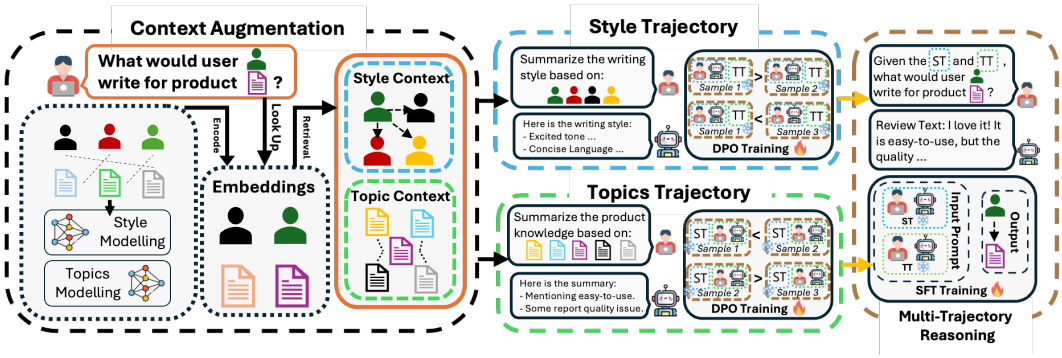



Figure 2: Overview of the proposed PAT framework. Yellow arrow represents one forward iteration.

### 3.1 Context Augmentation

In cold-start scenarios where users have limited history, the sparse history $\mathcal{H}_u$ of user $u$ prevents the model from constructing effective user representations. To bridge this gap, leveraging auxiliary information from a neighborhood of related users has been widely adopted in recommendation systems (Lika et al., 2014) and recently discovered to improve personalization (Au et al., 2025).

To create the auxiliary information, we construct a user-topic bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{U} \cup \mathcal{T}$ consists of users $u \in \mathcal{U}$ and entities $t \in \mathcal{T}$. For every historical entry $(x_{u,n}, y_{u,n}) \in \mathcal{H}_u$, we associate the text $y_{u,n}$ with a specific topic $t$. An edge $e_{u,t} \in \mathcal{E}$ exists if user $u$ has written about topic $t$. For example, if $y$ is user product reviews, then $t$ is the associated product; if $y$ is user emails, then $t$ is the email topic. Subsequently, the auxiliary information for user $u$ is defined as, $\mathcal{A}_u = (\mathcal{C}_u^{style}, \mathcal{C}_u^{topic})$, where $\mathcal{C}_u^{style}$ is the style context and $\mathcal{C}_u^{topic}$ is the topic knowledge context. We will discuss their construction separately in the rest of this subsection.

#### 3.1.1 Stylistic Context Extraction

The style context $\mathcal{C}_u^{style}$ aims to capture the linguistic patterns and tone of a given user. In cold-start scenarios, where $|\mathcal{H}_u|$ is small, the model cannot construct a reliable representation from the user's own history alone. To overcome this, we leverage graph learning to propagate stylistic cues across the user-topic bipartite graph $\mathcal{G}$. Let $\phi(\cdot)$ be a semantic encoder that maps text to a continuous embedding space. Since our goal is to propagate *stylistic* information rather than topical semantics, we adopt a content-independent style encoder (Wegmann et al., 2022).

For each user node $u \in \mathcal{U}$, we construct the initial embedding by aggregating all prior texts written by $u$ in their historical profile. Concretely, this is obtained by mean-pooling the style embeddings of all associated texts: $\mathbf{h}_u^{(0)} = \frac{1}{|\mathcal{H}_u|} \sum_{(x_{u,n}, y_{u,n}) \in \mathcal{H}_u} \phi(y_{u,n})$. For $|\mathcal{H}_u| = 0$, we use $\mathbf{h}_u^{(0)} = \phi(x_{\text{target}})$. Similarly, for each topic node $t \in \mathcal{T}$, let $\mathcal{R}_t$ be all texts written for topic $t$ across users. Its representation is obtained by mean-pooling the corresponding style embeddings: $\mathbf{h}_t^{(0)} = \frac{1}{|\mathcal{R}_t|} \sum_{y \in \mathcal{R}_t} \phi(y)$.

To capture higher-order structural dependencies between users and entities, we apply a GraphSAGE encoder (Hamilton et al., 2018) over the graph $\mathcal{G}$ and obtain the final user embedding $\mathbf{h}_u = \mathbf{h}_u^{(L)}$ a after $L$ propagation layers. We then retrieve the stylistically similar users for $u$ by computing cosine similarity of $\mathbf{h}_u$ with all other users:

$$\mathcal{N}_u^{style} = \underset{v \in \mathcal{U} \setminus \{u\}}{\arg\,\text{topk}_1} \frac{\mathbf{h}_u \cdot \mathbf{h}_v}{\|\mathbf{h}_u\| \, \|\mathbf{h}_v\|}, \tag{2}$$

where $\arg\,\text{topk}_1$ returns the set of $k_1$ (hyperparameter) users with the highest cosine similarity.

The stylistic context is finally constructed by collecting historical texts from these neighbors:

$$\mathcal{C}_u^{style} = \{y_{v,j} \mid v \in \mathcal{N}_u^{style}\}, \tag{3}$$

which provides the model with style-consistent exemplars independent of the target topic.

### 3.1.2 TOPIC KNOWLEDGE CONTEXT EXTRACTION

While stylistic context captures *how* a user tends to write, effective personalization also requires grounding the generation in *what* should be written—namely, topic-specific knowledge and sentiment. The topic knowledge context $\mathcal{C}_u^{topic}$ is therefore designed to retrieve writings that are semantically aligned with the target topic.

Given a target topic $t_{\text{target}}$, we first identify all writings associated with the specific topic, forming an initial candidate set $\mathcal{Y}_{\text{cand}}(t_{\text{target}})$. However, exact topic matches can be sparse in real-world applications. To improve robustness, we introduce a semantic backoff mechanism at the topic level. Each topic is represented by a pooled embedding of its associated texts, and cosine similarity is used to retrieve a small set of semantically similar topics when insufficient exact matches are available.

To align retrieved knowledge with the target user's perspective, we define $\mathcal{N}_u^{topic}$ analogously to $\mathcal{N}_u^{style}$ using cosine similarity in the GraphSAGE embedding space, while restricting candidates to $\mathcal{U}_{\text{cand}}$, the users who have authored writings in the candidate topic set $\mathcal{Y}_{\text{cand}}$. We retain the top-$k_2$ such users as knowledge context and form $\mathcal{C}_u^{topic}$ from their writings on the target topic, i.e., $\mathcal{C}_u^{topic} = \{y_{v,t_{\text{target}}} \mid v \in \mathcal{N}_u^{topic}\}$. Further details on graph training and zero-history inference are provided in Appendix A.5.

## 3.2 MULTI-TRAJECTORY REASONING

While the context augmentation stage constructs $\mathcal{A}_u = (\mathcal{C}_u^{style}, \mathcal{C}_u^{topic})$, directly concatenating these raw neighbor texts into a single prompt is often noisy. For example, stylistic neighbors may contain irrelevant cues, and topic knowledge context may exhibit misaligned preferences. To robustly integrate these heterogeneous signals, PAT employs a multi-trajectory reasoning mechanism that decomposes personalization into two complementary reasoning trajectories—(i) a *style trajectory* that extracts user-level writing patterns from $\mathcal{C}_u^{style}$, and (ii) an *topic knowledge trajectory* that extracts topic-specific knowledge and sentiment from $\mathcal{C}_u^{topic}$. The two trajectories are then fused to guide a downstream generation model.

**Trajectory Reasoning Agents.** We instantiate two reasoning agents parameterized by LLMs: a style reasoning agent $\pi_{\theta_s}$ and an topic knowledge reasoning agent $\pi_{\theta_t}$. Given the retrieved neighbor texts, the style agent takes the context $\mathcal{C}_u^{style}$ together with the user's review history $\mathcal{H}_u$, and produces a style summary as the reasoning trajectory: $s_u = \pi_{\theta_s}(\mathcal{P}_{style}(\mathcal{C}_u^{style}, \mathcal{H}_u))$, where $s_u$ denotes the summarized stylistic attributes (e.g., tone, phrasing, typical structure), and $\mathcal{P}_{style}$ denotes the prompt construction function that formats the style context and user history into the LLM input sequence.

Similarly, the topic agent takes the topic knowledge context and produces a knowledge summary.

$$p_{u,t_{\text{target}}} = \pi_{\theta_t}(\mathcal{P}_{topic}(\mathcal{C}_u^{topic})) \tag{4}$$

where $p_{u,t_{\text{target}}}$ captures topic context relevant to $t_{\text{target}}$ and $\mathcal{P}_{topic}$ denotes the prompt construction function that formulates the topic knowledge context into the LLM prompt.

**Generation with Fused Trajectories.** We then condition a generation model $\pi_{\theta_g}$ on both summarized trajectories (and optional original auxiliary texts) to produce the output for the target topic:

$$\hat{y} = \pi_{\theta_g}(\mathcal{P}_{gen}(x_{target}, s_u, p_{u,t_{\text{target}}}, \mathcal{C}_u^{style}, \mathcal{C}_u^{topic})) \tag{5}$$

where $\mathcal{P}_{gen}$ denotes the prompt construction function for final personalized text generation.

**Optimization via Differential Rewards.** A key challenge is that the generation of the intermediate trajectory summaries cannot be directly supervised: there is no gold label for $s_u$ or $p_{u,t_{\text{target}}}$. To address this, we adopt an iterative self-improvement procedure based on differential reward signals (Wang et al., 2021; Li et al., 2025). At a high level, the trajectory reasoning agents are optimized by measuring how alternative trajectories affect the downstream generation quality, and we update the agent to prefer the better-performing trajectories. We visualize the method's computation graph in Figure 3.

As shown in the figure, in one forward pass, $\pi_{\theta_g}$ takes the input from $\pi_{\theta_s}$ and $\pi_{\theta_t}$. We thus aim to optimize the trajectory agents $\pi_{\theta_s}$ and $\pi_{\theta_t}$ such that their intermediate outputs $s_u$ and $p_{u,t_{\text{target}}}$ induce

higher-quality downstream generations $\hat{y}$. However, directly differentiating the generation loss with respect to the trajectory agents is intractable because of the discrete nature.

As a result, we leverage *differential rewards* to assign credit to trajectory-level decisions. Concretely, we sample multiple candidate summaries from each trajectory agent and roll them out through the generation model, obtaining a set of outputs whose relative task rewards reflect the quality of the underlying trajectories. Based on the relative rewards, we can use preference optimization (Rafailov et al., 2024) to update the underlying trajectory agent.

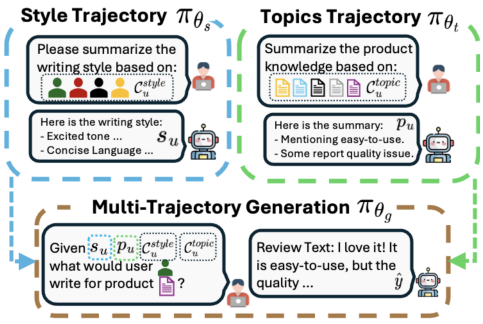



Figure 3: Computation graph of PAT.

Specifically, for the style reasoning agent, we sample $M_1$ candidate summaries:

$$\{s_u^{(m)}\}_{m=1}^{M_1} \sim \pi_{\theta_s}\big(\mathcal{P}_{\text{style}}(\mathcal{C}_u^{style}, \mathcal{H}_u)\big)\,. \quad (6)$$

To evaluate the quality of each candidate summary, we fix the topic trajectory $p_u$ and roll out each $s_u^{(m)}$ through the downstream generation model, producing a generated output

$$\hat{y}^{(m)} = \pi_{\theta_g}\Big(x_{\text{target}},\ s_u^{(m)},\ p_{u,t_{\text{target}}},\ \mathcal{C}_u^{style},\ \mathcal{C}_u^{topic}\Big)\,. \quad (7)$$

We then compute a task-specific reward by comparing the generated output $\hat{y}^{(m)}$ with the ground-truth target $y$: $R^{(m)} = R\big(\hat{y}^{(m)}, y\big)$, where $R(\cdot,\cdot)$ denotes the downstream evaluation function (e.g., ROUGE-based similarity for text generation).

Based on the relative rewards $\{R^{(m)}\}$, we rank the sampled trajectory summaries and construct a preference dataset for the style reasoning agent. Concretely, for any pair of summaries $(s_u^{(i)}, s_u^{(j)})$ such that $R^{(i)} > R^{(j)}$, we treat $s_u^{(i)}$ as a preferred trajectory over $s_u^{(j)}$ and add the corresponding pairwise preference to the training set. These preference pairs are then used to update the style reasoning agent via Direct Preference Optimization (DPO).

Similarly, we optimize the topic reasoning agent by sampling $M_2$ candidate topic-aware summaries:

$$\{p_{u,t_{\text{target}}}^{(m)}\}_{m=1}^{M_2} \sim \pi_{\theta_t}\big(\mathcal{P}_{topic}(\mathcal{C}_u^{topic})\big)\,. \quad (8)$$

For each candidate $p_{u,t_{\text{target}}}^{(m)}$, we fix the style trajectory $s_u$ and roll out the generation model, producing:

$$\hat{y}^{(m)} = \pi_{\theta_g}\Big(x_{\text{target}},\ s_u,\ p_{u,t_{\text{target}}}^{(m)},\ \mathcal{C}_u^{style},\ \mathcal{C}_u^{topic}\Big)\,. \quad (9)$$

We then compute the corresponding task reward $R^{(m)} = R(\hat{y}^{(m)}, y)$ by comparing the generated output with the ground-truth target. The knowledge trajectory reasoning agent is again optimized through DPO on the ranked trajectories based on the rewards.

Finally, to update the generation model $\pi_{\theta_g}$ to optimally utilize the refined trajectory summaries, we adopt a supervised fine-tuning (SFT) objective. Specifically, we construct training examples by conditioning the generator on the reasoning trajectories and the retrieved contexts, and train it to reproduce the ground-truth target output $y$. The generation model is optimized by minimizing the negative log-likelihood:

$$\min_{\theta_g} \mathbb{E}_{(x_{\text{target}}, s_u, p_{u,t_{\text{target}}}, y)}\Big[-\log \pi_{\theta_g}\Big(y \mid x_{\text{target}},\ s_u,\ p_{u,t_{\text{target}}},\ \mathcal{C}_u^{style},\ \mathcal{C}_u^{topic}\Big)\Big]. \quad (10)$$

This step is crucial in aligning the generator with the reasoning trajectories, further improving personalization.

**Iterative Optimization.** It is important to note that the trajectory reasoning agents and generation model are inherently interdependent. Specifically, during trajectory sampling, the rollout in Eq. (7) evaluates candidate style summaries by conditioning on the current generation model $\pi_{\theta_g}$ and a fixed topic summary produced by $\pi_{\theta_t}$, while Eq. (9) analogously evaluates topic-aware summaries in conjunction with $\pi_{\theta_g}$ and the style agent $\pi_{\theta_s}$. Conversely, the supervised fine-tuning objective in Eq. (10) updates the generation model by conditioning its outputs on the trajectory summaries produced by $\pi_{\theta_s}$ and $\pi_{\theta_t}$.

To account for this interdependence, PAT adopts an iterative optimization strategy that alternates between trajectory refinement and generator adaptation. At each iteration, we first optimize the style reasoning agent $\pi_{\theta_s}$ and the topic knowledge reasoning agent $\pi_{\theta_t}$ using differential reward signals derived from downstream generation quality, implemented via Direct Preference Optimization. Given the updated trajectory agents, we then update the generation model $\pi_{\theta_g}$ through supervised fine-tuning using the selected high-reward trajectory summaries.

**Algorithm 1:** PAT: Multi-Trajectory Self-Improvement with Differential Rewards for Personalized Text Generation.

**Input:** Training data $\mathcal{D}$; iterations $T$; samples $M_1$, $M_2$; neighbor sizes $k_1, k_2$
**Output:** Trajectory agents $\pi_{\theta_s}, \pi_{\theta_t}$ and generator $\pi_{\theta_g}$
**Precompute contexts.** User–topic graph $\mathcal{G}$ and node embeddings; $\mathcal{C}_u^{style}$ (top-$k_1$ stylistic neighbors) and $\mathcal{C}_u^{topic}$ (top-$k_2$ topic-aligned neighbors)
**for** $t = 1$ **to** $T$ **do**
  `// (1) Update style agent via differential rewards`
  Sample $\{s_u^{(m)}\}_{m=1}^{M_1}$ from $\pi_{\theta_s}(\mathcal{P}_{style}(\mathcal{C}_u^{style}, \mathcal{H}_u))$
  Roll out each $s_u^{(m)}$ through $\pi_{\theta_g}$ (fixing $p_u$) and compute rewards $R^{(m)}$
  Build pairwise preferences from reward ranking; update $\pi_{\theta_s}$ with DPO
  `// (2) Update topic agent via differential rewards`
  Sample $\{p_u^{(m)}\}_{m=1}^{M_2}$ from $\pi_{\theta_t}(\mathcal{P}_{topic}(\mathcal{C}_u^{topic}))$
  Roll out each $p_u^{(m)}$ through $\pi_{\theta_g}$ (fixing $s_u$) and compute rewards $R^{(m)}$
  Build pairwise preferences from reward ranking; update $\pi_{\theta_t}$ with DPO
  `// (3) Update generator on silver trajectories`
  Select silver summaries $(s_u^\star, p_u^\star)$; SFT $\pi_{\theta_g}$ on $(x_{\text{target}}, s_u^\star, p_u^\star, y)$
**return** $\pi_{\theta_s}, \pi_{\theta_t}, \pi_{\theta_g}$

This alternating optimization progressively improves both the quality of intermediate reasoning trajectories and the generator's ability to effectively exploit them. By propagating downstream task feedback to trajectory-level decisions through differential rewards, PAT mitigates the absence of direct supervision for intermediate summaries and enables stable self-improvement, especially for cold-start users (i.e., sparse or noisy history). The pseudo-code of PAT is provided in Algorithm 1.

# 4 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate the performance of PAT on the personalized text generation, across settings including short text generation and long text generation. In addition, we also include ablation studies that explores the individual components proposed in PAT.

## 4.1 DATASETS AND METRICS

Our experimental design follows established evaluation protocols in prior studies (Au et al., 2025) and is conducted on three commonly used benchmarks: Amazon Reviews (Ni & McAuley, 2018), Hotel Reviews (Kanouchi et al., 2020), and Stylized Feedback (Alhafni et al., 2024). All three datasets consist of large-scale human-authored texts and exhibit a pronounced cold-start setting, with over 95% of users having fewer than two historical writings. The dataset statistics are presented in Appendix A.1. We evaluate performance on long-form text generation and short-form text generation. In the long-form generation task, the model is conditioned on a title prompt, while the short-form generation task involves summarizing a given paragraph.

For both generation tasks, we report standard lexical similarity metrics, including ROUGE-1, ROUGE-L, and METEOR, consistent with prior work (Au et al., 2025; Kumar et al., 2024; Salemi et al., 2024). Since such metrics primarily capture surface-level overlap, we additionally employ an LLM-as-a-Judge framework (Liu et al., 2023), in which a high-capacity language model performs pairwise evaluations of personalization quality. Details of the LLM-as-a-Judge protocol are provided in Appendix E. A comprehensive description of the evaluation metrics and experimental configuration is included in Appendix A. Additional experiments on efficiency are supplied in Appendix B.

Table 2: Results across benchmarks for long- and short-text generation. Avg.R denotes the average rank across all benchmarks and metrics; lower is better.

LONG TEXT GENERATION

| Method | LLM | Amazon | | | | Hotel | | | | Feedback | | | | Avg.R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | R-1 | R-L | MET | LLM | R-1 | R-L | MET | LLM | R-1 | R-L | MET | LLM | |
| **PAT** | *Qwen3* | **0.210** | **0.159** | 0.176 | 3.432 | 0.240 | 0.142 | **0.194** | **3.438** | **0.242** | **0.179** | **0.210** | 3.357 | 1.50 |
| | *LlaMA3* | **0.197** | **0.151** | **0.158** | 3.100 | **0.277** | **0.167** | **0.204** | **3.231** | **0.231** | **0.175** | **0.179** | 3.171 | 1.25 |
| GraSPeR | *Qwen3* | 0.168 | 0.129 | 0.150 | 2.961 | 0.234 | **0.156** | 0.163 | 2.750 | 0.196 | 0.149 | 0.160 | 3.040 | 3.21 |
| | *LlaMA3* | 0.167 | 0.141 | 0.124 | 3.006 | 0.211 | 0.148 | 0.152 | 2.520 | 0.191 | 0.154 | 0.134 | 2.930 | 3.00 |
| PGraph | *Qwen3* | 0.183 | 0.132 | **0.177** | **3.725** | **0.254** | 0.148 | 0.193 | 3.174 | 0.213 | 0.153 | 0.191 | **3.685** | 1.67 |
| | *LlaMA3* | 0.152 | 0.133 | 0.109 | 3.399 | 0.260 | 0.155 | 0.182 | 1.838 | 0.211 | 0.152 | 0.175 | **3.290** | 2.58 |
| LaMP | *Qwen3* | 0.168 | 0.122 | 0.158 | 3.353 | 0.178 | 0.117 | 0.114 | 2.733 | 0.179 | 0.124 | 0.161 | 3.107 | 3.62 |
| | *LlaMA3* | 0.161 | 0.114 | 0.149 | **3.701** | 0.224 | 0.141 | 0.148 | 2.651 | 0.181 | 0.122 | 0.164 | 2.873 | 3.17 |

SHORT TEXT GENERATION

| Method | LLM | Amazon | | | | Hotel | | | | Feedback | | | | Avg.R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | R-1 | R-L | MET | LLM | R-1 | R-L | MET | LLM | R-1 | R-L | MET | LLM | |
| **PAT** | *Qwen3* | **0.177** | **0.171** | **0.155** | **3.744** | 0.133 | 0.123 | **0.118** | **3.739** | **0.191** | **0.189** | **0.174** | **3.716** | 1.21 |
| | *LlaMA3* | **0.150** | **0.147** | 0.116 | **3.261** | **0.126** | **0.121** | 0.085 | **3.462** | **0.166** | **0.163** | 0.116 | **3.325** | 1.75 |
| GraSPeR | *Qwen3* | 0.158 | 0.155 | 0.134 | 3.388 | **0.150** | **0.145** | **0.118** | 3.282 | 0.170 | 0.168 | 0.127 | 3.292 | 2.46 |
| | *LlaMA3* | 0.132 | 0.129 | **0.128** | 3.212 | 0.120 | 0.117 | 0.108 | 3.087 | 0.157 | 0.155 | 0.127 | 3.175 | 2.17 |
| PGraph | *Qwen3* | 0.124 | 0.120 | 0.132 | 3.722 | 0.122 | 0.114 | 0.106 | 3.651 | 0.127 | 0.122 | 0.134 | 3.586 | 2.75 |
| | *LlaMA3* | 0.120 | 0.115 | 0.121 | 3.141 | 0.125 | 0.116 | **0.111** | 3.196 | 0.139 | 0.134 | **0.139** | 3.076 | 2.50 |
| LaMP | *Qwen3* | 0.107 | 0.102 | 0.107 | 3.460 | 0.112 | 0.104 | 0.096 | 3.512 | 0.115 | 0.110 | 0.135 | 3.356 | 3.58 |
| | *LlaMA3* | 0.115 | 0.109 | 0.124 | 3.081 | 0.104 | 0.096 | 0.100 | 3.051 | 0.121 | 0.116 | 0.136 | 2.888 | 3.58 |

### 4.2 BASELINES

We benchmark against three state-of-the-art personalization baselines. **LaMP** (Salemi et al., 2024) conditions on a user's past writing via prompts. **PGraphRAG** (Au et al., 2025) (i.e., PGraph) employs graph-based retrieval augmented generation with BM25 for personalization but lacks reasoning or fine-tuning. **GraSPeR** (**?**) augments sparse users' context with graph learning, and leverages reasoning to refine the outputs. We evaluate all baselines with Qwen3-4B-Instruct (Bai et al., 2023) and Llama3.2-3B-Instruct (Grattafiori et al., 2024).

### 4.3 MAIN RESULTS

We present the main experimental results in Table 2 for the Amazon Review, Hotel Experience, and Stylized Feedback benchmarks. PAT generally outperforms the baselines on long-text generation across ROUGE-1, ROUGE-L, and METEOR, while remaining competitive on short-text tasks with smaller relative gains. This pattern suggests that iterative trajectory refinement is especially beneficial for longer outputs, which require sustained coherence and integration of user- and topic-specific signals, whereas shorter outputs may require less complex personalization.

Moreover, PAT demonstrates clear advantages under LLM-as-a-Judge evaluations. Across all three datasets, PAT achieves top (or near-top) LLM-as-a-Judge scores. This suggests the improvements of PAT extend beyond n-gram overlap and translate into higher-quality, more preference-aligned outputs as judged by strong language models. Additional baseline comparisons, paired significance tests, and a small-scale human validation of the LLM judge are provided in Appendix B.

### 4.4 TRAJECTORY SAMPLING BUDGET

The differential reward mechanism in PAT relies on a pool of candidate trajectories from which pairwise preferences are constructed for DPO. We set the default sampling budget to $M_1 = M_2 = 3$, which represents the smallest pool that supports meaningful pairwise comparisons while keeping the rollout cost bounded. To verify that this default does not under represent the preference signal, we increase the budget to $M_1 = M_2 = 5$ and rerun PAT on Amazon Short Text Generation with the Qwen3 backbone. As

Table 3: Sensitivity to the trajectory sampling budget on Amazon Short Text Generation with the Qwen3 backbone. Increasing the budget from 3 to 5 yields only marginal improvements, indicating that PAT captures sufficient preference signal at $M_1 = M_2 = 3$.

| $M_1 = M_2$ | R-1 | R-L | MET | LLM |
|---|---|---|---|---|
| 3 (default) | 0.177 | 0.171 | 0.155 | 3.744 |
| 5 | 0.180 | 0.174 | 0.165 | 3.745 |

shown in Table 3, the gains from this larger budget are marginal across all four metrics, indicating that PAT already captures the bulk of the available preference signal at the default budget.

### 4.5 Model Convergence

We also investigate the learning dynamics and stability of our approach; we track the convergence behavior of PAT by plotting the average performance metrics for both text and title generation on the Amazon Review dataset across training iterations, as illustrated in Figure 4.

We observe that as iteration count increases, model performance exhibits a general upward trend across all primary metrics. Specifically, they show a steep initial improvement, then a gradual stabilization, suggesting PAT efficiently converges to a high-performing local optimum.




Figure 4: Text generation performance as iteration number increases on the Amazon Review dataset.

**Note on the training epochs.** To ensure that these performance gains are a result of the iterative refinement process rather than simply an artifact of increased training budget, within each iteration, we cap both the Supervised Fine-Tuning and Direct Preference Optimization at 50 steps. This ensures that each individual iteration reaches a point of local convergence independently. Consequently, the observed improvements across successive iterations can be attributed to the quality of the generated trajectories and the iterative feedback loop, rather than training time.

### 4.6 Performance Analysis by History Length

Table 4 evaluates PAT across user history lengths. Here, *zero history* means no additional past writings in the user's profile; the current task input remains available. We initialize the user embedding from this input (the review title for long-text generation or the review text for short-text generation) and incorporate neighborhood information through the user's connection to the target topic. With zero history, PAT outperforms PGraph by an average of $24.81\%$ across metrics. The gains decrease to $7.10\%$ and $1.68\%$ with one and two historical writings, respectively, suggesting that auxiliary context is most beneficial when personal history is sparse.

Table 4: Comparison between PAT and PGraph with Avg. Improvement on the Short Text Generation for the Amazon dataset.

| # Hist. | Method | ROUGE-1 | ROUGE-L | METEOR | Δ% |
|---|---|---|---|---|---|
| 0 | PAT | 20.73 | 15.82 | 17.69 | +24.81% |
| | PGraph | 16.23 | 12.53 | 14.69 | |
| 1 | PAT | 23.10 | 17.63 | 18.58 | +7.10% |
| | PGraph | 21.39 | 16.30 | 17.67 | |
| 2 | PAT | 21.61 | 14.23 | 14.88 | +1.68% |
| | PGraph | 20.46 | 12.98 | 16.56 | |

## 5 Ablation Study

In this section, We ablate the style and topic trajectories and compare iterative DPO/SFT training with a zero-shot variant.

**Impact of Multi-Trajectory Reasoning** To evaluate PAT's model architecture, we compare the full PAT against several variants that selectively remove the reasoning trajectories. As shown in Table 5, removing either the topic knowledge trajectory (*w/o Knowledge Trajectory*) or the style trajectory (*w/o Style Trajectory*) leads to a consistent decline in performance across all metrics. Furthermore, the most significant degradation occurs in the case when both trajectories are excluded, where the model is provided only with raw retrieved information $\mathcal{A}_u$.

Table 5: Ablation study of PAT highlighting the contribution of style and product trajectories. We report ROUGE-1 (R-1), ROUGE-L (R-L), METEOR, and LLM-as-a-Judge (LLM) scores.

| Variant | R-1 | R-L | MET | LLM |
|---|---|---|---|---|
| PAT | **0.18** | **0.17** | **0.16** | **3.75** |
| w/o Style Trajectory | 0.16 | 0.16 | 0.15 | 3.54 |
| w/o Knowledge Trajectory | 0.17 | 0.16 | 0.16 | 3.58 |
| w/o Both Trajectories | 0.13 | 0.12 | 0.12 | 3.24 |
| w/o Training (Zero-shot) | 0.12 | 0.11 | 0.13 | 3.35 |

**Impact of Iterative Training** Here we evaluate PAT in a zero-shot configuration, where we utilize the proposed retrieval and reasoning architecture but replace the fine-tuned agents ($\pi_{\theta_s}, \pi_{\theta_t}, \pi_{\theta_g}$) with their base LLM counterparts. As illustrated in Table 5, the "w/o Training (Zero-shot)" variant

yields the lowest performance across the board. While the retrieval-augmented framework provides the model with the necessary contextual knowledge, the base LLM lacks the alignment required to prioritize the specific stylistic cues and domain knowledge that lead to high-quality personalization. By leveraging differential rewards and DPO, PAT learns to prefer reasoning trajectories that improve downstream generation quality.

## 6 Case Study

To validate the efficacy of our multi-trajectory reasoning, we present a qualitative analysis of the intermediate reasoning traces and generated outputs in Table 6. In Iteration 0, the model fails to filter noise from the raw retrieved context; the Style Trace remains generic, while the Topic Trace hallucinates negative attributes. Consequently, the baseline generation resorts to a safe, generic review that lacks user-specific character. In contrast, after optimization with differential data rewards, the Topic Knowledge Trajectory successfully isolates accurate product details (vibrant red color), and Style Trajectory captures more granular linguistic characteristics, e.g., *emphatic*, resulting in the better personalized text generation.

Table 6: Case study. Orange indicates overly general traces, red indicates traces that do not help generation, and green indicates reasoning that directly supports correct personalization.

| | Iteration 0 (Baseline) | Iteration 4 (Refined) |
|---|---|---|
| Ground Truth | *Target Review:* "Love this! Beautiful red. Stays on all day!" | |
| Style Trace | Informal tone; Positive language; Focus on product experiences and personal preferences; Emphasis on product quality. | Clumsy tone and short sentences; Use of emphatic language (e.g. Wasted my money!); Use of colloquial expression. |
| Topic Trace | Brand: Maybelline, Quality: Poor, Color: Red, Staying Power: Good. | The product is a lipstick with a vibrant red color that stays in place well. Considered affordable. |
| Final Text | I love this lipstick. It is a great color and it is so easy to apply. It is long lasting and it does not bleed. I will definitely be buying more. | I love the color and the long lasting finish! |

## 7 Related Work

**LLM Personalization.** LLM personalization follows two main directions. First, personalized preference optimization (Li et al., 2024; Jang et al., 2023; Wu et al., 2023) aligns models with user behavior through user embeddings or parameter updates. To capture individual rather than general preferences, approaches model multiple preference dimensions and personalize reward weights (Wu et al., 2023) or preference assignment (Jang et al., 2023). More recent work learns user embeddings directly from data (Li et al., 2024) and constructs user-specific reward functions (Poddar et al., 2024). Second, prompt-based personalization steers generation by supplying personal context in the input prompt (Salemi et al., 2024; Kumar et al., 2024; Au et al., 2025; **?**). Early work retrieves personal context (Salemi et al., 2024; Kumar et al., 2024), while recent methods address cold-start settings (**?**Au et al., 2025).

**Differential Rewards.** Traditional Reinforcement Learning from Human Feedback (RLHF) optimizes a single scalar reward for shared preferences, which can obscure minority viewpoints or conflicting objectives. Differential reward methods decompose this signal to capture fine-grained or preferences. Multi-objective reward models use separate heads for distinct attributes, enabling trade-offs or user-steerable inference (Wu et al., 2023; Dong et al., 2023). Group-conditional rewards address majority bias and demographic or ideological diversity (Bakker et al., 2022). Contrastive reward frameworks model differences between user-specific rewards and a population baseline, enabling efficient adaptation without extensive retraining (Poddar et al., 2024; Sorensen et al., 2024; 2025). RAG-DDR (Li et al., 2025) instead uses downstream generation feedback for model optimization.

## 8 Conclusion

In this paper, we presented PAT, a framework designed to tackle cold-start LLM personalization by decomposing user context into complementary writing-style and topic-knowledge trajectories. By leveraging an iterative dual-reasoning mechanism optimized via differential rewards, PAT effectively synthesizes noisy neighbor signals to align generation with user preferences, even when personal history is sparse. Our extensive experiments demonstrate that PAT significantly outperforms state-of-the-art baselines in sparse-data scenarios, achieving an average improvement of over 15% for users with zero history.

## 9 AI USE DISCLOSURE

In this work, we used generative AI tools for brainstorming research idea, implementing part of the codebase, and do preliminary analysis on the experiment result. We have not used generative AI tools for autoresearch, propose hypothesis, or provide feedback on methodology, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for draft part of the research paper, create or modify the teaser figure and images, and summarize existing landscape in the research area that is within our interests. We have reviewed all AI-assisted work. We checked LLM-generated research ideas for potential plagiarism through a manual literature survey, and audited the claims and code written by LLMs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

## REFERENCES

Bashar Alhafni, Vivek Kulkarni, Dhruv Kumar, and Vipul Raheja. Personalized text generation with fine-grained linguistic control. In Ameet Deshpande, EunJeong Hwang, Vishvak Murahari, Joon Sung Park, Diyi Yang, Ashish Sabharwal, Karthik Narasimhan, and Ashwin Kalyan (eds.), *Proceedings of the 1st Workshop on Personalization of Generative AI Systems (PERSONALIZE 2024)*, pp. 88–101, St. Julians, Malta, March 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.personalize-1.8.

Anthropic. Claude 4.5 sonnet. Generative AI Chatbot, sep 2026. URL https://claude.ai/. Accessed: 2026-09-26. Prompt: "[Your Prompt Here]".

Steven Au, Cameron J. Dimacali, Ojasmitha Pedirappagari, Namyong Park, Franck Dernoncourt, Yu Wang, Nikos Kanakaris, Hanieh Deilamsalehy, Ryan A. Rossi, and Nesreen K. Ahmed. Personalized graph-based retrieval for large language models, 2025. URL https://arxiv.org/abs/2501.02157.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report, 2023. URL https://arxiv.org/abs/2309.16609.

Michiel A. Bakker, Martin J. Chadwick, Hannah R. Sheahan, Michael Henry Tessler, Lucy Campbell-Gillingham, Jan Balaguer, Nat McAleese, Amelia Glaese, John Aslanides, Matthew M. Botvinick, and Christopher Summerfield. Fine-tuning language models to find agreement among humans with diverse preferences, 2022. URL https://arxiv.org/abs/2211.15006.

Yi Dong, Zhilin Wang, Makesh Narsimhan Sreedhar, Xianchao Wu, and Oleksii Kuchaiev. Steerlm: Attribute conditioned sft as an (user-steerable) alternative to rlhf, 2023. URL https://arxiv.org/abs/2310.05344.

Aaron Grattafiori et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs, 2018. URL https://arxiv.org/abs/1706.02216.

Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*, 2024.

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, 2023. URL https://arxiv.org/abs/2310.11564.

Shin Kanouchi, Masato Neishi, Yuta Hayashibe, Hiroki Ouchi, and Naoaki Okazaki. You may like this hotel because ...: Identifying evidence for explainable recommendations. In Kam-Fai Wong, Kevin Knight, and Hua Wu (eds.), *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pp. 890–899, Suzhou, China, December 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.aacl-main.89. URL https://aclanthology.org/2020.aacl-main.89.

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.

Xinyu Li, Ruiyang Zhou, Zachary C. Lipton, and Liu Leqi. Personalized language modeling from personalized human feedback, 2024. URL https://arxiv.org/abs/2402.05133.

Xinze Li, Sen Mei, Zhenghao Liu, Yukun Yan, Shuo Wang, Shi Yu, Zheni Zeng, Hao Chen, Ge Yu, Zhiyuan Liu, Maosong Sun, and Chenyan Xiong. Rag-ddr: Optimizing retrieval-augmented generation using differentiable data rewards, 2025. URL https://arxiv.org/abs/2410.13509.

Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4, Part 2):2065–2073, 2014. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2013.09.005. URL https://www.sciencedirect.com/science/article/pii/S0957417413007240.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: NLG evaluation using gpt-4 with better human alignment. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.153. URL https://aclanthology.org/2023.emnlp-main.153/.

Durga Prasad Maram, Dhruvin Gandhi, Zonghai Yao, Gayathri Akkinapalli, Franck Dernoncourt, Yu Wang, Ryan A. Rossi, and Nesreen K. Ahmed. Iterative critique-refine framework for enhancing llm personalization, 2025. URL https://arxiv.org/abs/2510.24469.

Jianmo Ni and Julian McAuley. Personalized review generation by expanding phrases and attending on aspect-aware representations. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 706–711, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2112. URL https://aclanthology.org/P18-2112/.

Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. Personalizing reinforcement learning from human feedback with variational preference learning, 2024. URL https://arxiv.org/abs/2408.10075.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

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.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. Lamp: When large language models meet personalization, 2024. URL https://arxiv.org/abs/2304.11406.

Alireza Salemi, Cheng Li, Mingyang Zhang, Qiaozhu Mei, Weize Kong, Tao Chen, Zhuowan Li, Michael Bendersky, and Hamed Zamani. Reasoning-enhanced self-training for long-form personalized text generation, 2025. URL https://arxiv.org/abs/2501.04167.

Taylor Sorensen, Jared Moore, Jillian Fisher, Mitchell Gordon, Niloofar Mireshghallah, Christopher Michael Rytting, Andre Ye, Liwei Jiang, Ximing Lu, Nouha Dziri, Tim Althoff, and Yejin Choi. A roadmap to pluralistic alignment, 2024. URL https://arxiv.org/abs/2402.05070.

Taylor Sorensen, Pushkar Mishra, Roma Patel, Michael Henry Tessler, Michiel A. Bakker, Georgina Evans, Iason Gabriel, Noah Goodman, and Verena Rieser. Value profiles for encoding human variation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 2047–2095, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.106. URL https://aclanthology.org/2025.emnlp-main.106/.

Xinyi Wang, Hieu Pham, Paul Michel, Antonios Anastasopoulos, Jaime Carbonell, and Graham Neubig. Optimizing data usage via differentiable rewards, 2021. URL https://arxiv.org/abs/1911.10088.

Anna Wegmann, Marijn Schraagen, and Dong Nguyen. Same author or just same topic? towards content-independent style representations, 2022. URL https://arxiv.org/abs/2204.04907.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A. Smith, Mari Ostendorf, and Hannaneh Hajishirzi. Fine-grained human feedback gives better rewards for language model training, 2023. URL https://arxiv.org/abs/2306.01693.

Weizhi Zhang, Yuanchen Bei, Liangwei Yang, Henry Peng Zou, Peilin Zhou, Aiwei Liu, Yinghui Li, Hao Chen, Jianling Wang, Yu Wang, Feiran Huang, Sheng Zhou, Jiajun Bu, Allen Lin, James Caverlee, Fakhri Karray, Irwin King, and Philip S. Yu. Cold-start recommendation towards the era of large language models (llms): A comprehensive survey and roadmap, 2025a. URL https://arxiv.org/abs/2501.01945.

Zhehao Zhang, Ryan A. Rossi, Branislav Kveton, Yijia Shao, Diyi Yang, Hamed Zamani, Franck Dernoncourt, Joe Barrow, Tong Yu, Sungchul Kim, Ruiyi Zhang, Jiuxiang Gu, Tyler Derr, Hongjie Chen, Junda Wu, Xiang Chen, Zichao Wang, Subrata Mitra, Nedim Lipka, Nesreen Ahmed, and Yu Wang. Personalization of large language models: A survey, 2025b. URL https://arxiv.org/abs/2411.00027.

# A Additional Experimental Setup

This section provides dataset statistics, task definitions, evaluation details, and implementation settings.

## A.1 Datasets

We evaluate PAT on three datasets from the PGraphRAG benchmark (Au et al., 2025), covering different domains under sparse user histories.

**Amazon Review.** The Amazon Review dataset is constructed from the Amazon Review 2023 corpus (Hou et al., 2024). Its user-item graph contains users and products as nodes, with edges representing review interactions.

**Hotel Experience.** The Hotel Experience dataset is collected from the Datafiniti Hotel Reviews dataset (Au et al., 2025). Its user-hotel interactions are annotated with textual reviews.

**Stylized Feedback Review.** The Stylized Feedback Review dataset is derived from the Datafiniti Grammar and Online Product dataset (Au et al., 2025). It evaluates personalized feedback generation from user-product interactions.

Tables 7 and 8 report the dataset splits and task statistics, respectively. Hotel Experience has a smaller training set than the other datasets. Across all three datasets, the average profile contains approximately one review, reflecting the sparse-history setting.

Table 7: Dataset split sizes across training, validation, and test sets for the four domains.

| Dataset | Train Size | Validation Size | Test Size |
|---|---|---|---|
| User-Product Review | 20,000 | 2,500 | 2,500 |
| Stylized Feedback | 20,000 | 2,500 | 2,500 |
| Hotel Experiences | 9,000 | 2,500 | 2,500 |

Table 8: Data statistics for the PGraphRAG Benchmark across the four datasets. For each task, we report the average input and output lengths (in words), measured on the test set using BM25-based retrieval with GPT. The average profile size indicates the number of reviews per user used for personalization.

| Task | Type | Avg. Input Length | Avg. Output Length | Avg. Profile Size | # Classes |
|---|---|---|---|---|---|
| User-Product Review Generation | Long Text Generation | $3.754 \pm 2.71$ | $47.90 \pm 19.28$ | $1.05 \pm 0.31$ | - |
| Hotel Experiences Generation | Long Text Generation | $4.29 \pm 2.57$ | $76.26 \pm 22.39$ | $1.14 \pm 0.61$ | - |
| Stylized Feedback Generation | Long Text Generation | $3.35 \pm 2.02$ | $51.80 \pm 20.07$ | $1.09 \pm 0.47$ | - |
| User-Product Review Title Generation | Short Text Generation | $30.34 \pm 37.95$ | $7.02 \pm 1.14$ | $1.05 \pm 0.31$ | - |
| Hotel Experiences Summary Generation | Short Text Generation | $90.40 \pm 99.17$ | $7.64 \pm 0.92$ | $1.14 \pm 0.61$ | - |
| Stylized Feedback Title Generation | Short Text Generation | $37.42 \pm 38.17$ | $7.16 \pm 1.11$ | $1.09 \pm 0.47$ | - |

### A.2 TASKS

**Long Text Generation.** Given a review title and the available user context, the model generates a detailed review that reflects the user's writing style and preferences.

**Short Text Generation.** Given a review and the available user context, the model generates a concise review title or summary. This task requires selecting salient information while preserving personalized expression.

### A.3 METRICS

We evaluate lexical overlap using ROUGE-1, ROUGE-L, and METEOR, following the benchmark protocol (Au et al., 2025). We additionally use LLM-as-a-Judge evaluation to assess alignment between generated and reference texts. The judge prompt follows prior personalized-generation work (Salemi et al., 2025) and is provided in Appendix E. Appendix B.3 reports a small-scale human validation of this evaluator.

### A.4 EXPERIMENTAL CONFIGURATION

Experiments with the open-weight models were conducted on a single NVIDIA GeForce RTX 4090 GPU. We set the maximum number of training iterations to $T = 10$, retrieve $k_1 = k_2 = 5$ neighbors, and use trajectory sampling budgets of $M_1 = M_2 = 3$ during training. Hyperparameters are selected on the validation set before evaluation on the held-out test set.

### A.5 GRAPH TRAINING AND COLD-START INFERENCE

**Node representations and graph training.** The pretrained content-independent style encoder is frozen throughout training. User and topic features are initialized by mean-pooling embeddings of their associated available texts from the style encoder (Wegmann et al., 2022). GraphSAGE aggregation parameters are trained on the training subgraph using the unsupervised random-walk objective (Hamilton et al., 2018), following the PyTorch Geometric reference implementation without graph-specific tuning.

**Zero-history users.** Zero history means that a user has no additional past writings beyond the current task input. For long-text generation, the review title remains available; for short-text generation, the review text remains available. We initialize the user's feature from this conditioning text when historical writings are absent. The user is also connected to the target topic in the user-topic graph, allowing graph propagation to incorporate neighborhood information.

**Retrieval.** Style retrieval ranks candidate users by cosine similarity between user embeddings. Topic retrieval first selects writings associated with the target topic, with a semantic backoff over topic embeddings when necessary, and then ranks candidate authors by user similarity. These retrieval operations introduce no additional learned parameters.

**Training and evaluation separation.** GraphSAGE parameters are learned from the training subgraph, while DPO preference pairs and SFT labels are constructed from training targets. Validation and test users can appear as graph nodes through their topic connections. For each evaluation query, its target output is excluded from its conditioning evidence.

**Shared modules and new users.** The style agent, topic agent, and generator are shared across users within each dataset. Personalization is introduced through retrieved context rather than user-specific model parameters. New users can therefore be processed using the trained graph aggregator and language-model modules without per-user optimization.

# B Additional Experimental Results

## B.1 Additional Baseline Comparisons

We extend the Amazon short-text evaluation with PAG, which combines summarization and retrieval for personalization (Richardson et al., 2023), and PerFine, an iterative critique-refine framework (Maram et al., 2025). Both are implemented with the same Qwen backbone as PAT. We additionally evaluate Claude Sonnet 4.5 (Anthropic, 2026) using zero-shot prompting with the same retrieved context. These comparisons assess alternative personalization strategies and a stronger model, without equalizing training compute.

Table 9 reports the scores and corresponding pairwise $p$-values for these additional comparisons.

Table 9: Additional baseline comparisons on Amazon short-text generation. Higher scores are better for all metrics. Each block retains the PAT score and pairwise $p$-value reported for that comparison.

| Baseline | Metric | PAT | Baseline | Reported $p$ |
|---|---|---|---|---|
| PAG | ROUGE-1 | 0.1771 | 0.1196 | $4 \times 10^{-36}$ |
| | ROUGE-L | 0.1717 | 0.1152 | $2 \times 10^{-36}$ |
| | METEOR | 0.1553 | 0.1222 | $10^{-12}$ |
| | Judge | 3.744 | 3.242 | $8 \times 10^{-21}$ |
| PerFine | ROUGE-1 | 0.1771 | 0.1276 | $2 \times 10^{-27}$ |
| | ROUGE-L | 0.1717 | 0.1226 | $6 \times 10^{-28}$ |
| | METEOR | 0.1553 | 0.1236 | $10^{-12}$ |
| | Judge | 3.577 | 3.405 | $3 \times 10^{-7}$ |
| Sonnet 4.5 | ROUGE-1 | 0.1771 | 0.1606 | 0.0007 |
| | ROUGE-L | 0.1717 | 0.1553 | 0.0011 |
| | METEOR | 0.1553 | 0.1592 | 0.12 |
| | Judge | 3.577 | 3.644 | 0.044 |

PAT achieves higher scores than PAG and PerFine on all four metrics. Compared with Sonnet 4.5, it achieves higher ROUGE-1 and ROUGE-L, while Sonnet obtains higher METEOR and judge scores. The METEOR difference is not statistically significant at the nominal $0.05$ threshold ($p = 0.12$), whereas the judge difference favors Sonnet at this threshold ($p = 0.044$). These results indicate competitive performance from the smaller adapted model, with the relative advantage depending on the evaluation metric.

## B.2 Paired Statistical Tests

We conduct two-sided Wilcoxon signed-rank tests on paired per-example scores from the Amazon benchmark with the Qwen3 backbone, using $2{,}500$ examples per task. Table 10 reports comparisons with the original baselines. These tests assess differences between paired outputs from the evaluated models; they do not measure variability across independently trained runs.

Table 10: Two-sided paired Wilcoxon signed-rank $p$-values on Amazon with Qwen3 ($n = 2{,}500$ per task). Values below $10^{-300}$ are retained as reported bounds.

| | Short text | | | Long text | | |
|---|---|---|---|---|---|---|
| Baseline | R-1 | R-L | METEOR | R-1 | R-L | METEOR |
| LaMP | $2\times10^{-23}$ | $6\times10^{-30}$ | $2\times10^{-8}$ | $< 10^{-300}$ | $< 10^{-300}$ | $8\times10^{-278}$ |
| PGraphRAG | $3\times10^{-10}$ | $5\times10^{-14}$ | 0.69 | $3\times10^{-45}$ | $9\times10^{-76}$ | $10^{-24}$ |
| GraSPeR | 0.004 | 0.025 | 0.16 | $< 10^{-300}$ | $8\times10^{-107}$ | $< 10^{-300}$ |

At a nominal threshold of $0.05$, all reported comparisons are significant except short-text METEOR against PGraphRAG and GraSPeR. Statistical significance does not determine the direction or practical magnitude of a difference: for example, PGraphRAG has a slightly higher mean long-text

METEOR score than PAT on Amazon/Qwen3 in Table 2. We interpret these tests alongside the reported scores rather than as evidence of uniform superiority.

### B.3 Human Validation of the Judge

Three non-author annotators blindly evaluated 30 reference/generated-review pairs, seeing the same inputs as the LLM judge. Each item was assessed for reference match using the judge's seven-point criterion, style consistency, and content relevance. Table 11 reports the correlations between human assessments and judge scores. Individual-annotator Pearson correlations with the judge ranged from $0.54$ to $0.84$.

Table 11: Reported correlations between human ratings and LLM judge scores on 30 items evaluated by three non-author annotators.

| Human criterion | Pearson $r$ | Spearman $\rho$ | Reported $p$ |
|---|---|---|---|
| Reference match | 0.83 | 0.85 | $1.3 \times 10^{-4}$ |
| Style consistency | 0.76 | 0.76 | $10^{-3}$ |
| Content relevance | 0.79 | 0.76 | $5.2 \times 10^{-4}$ |

The positive correlations provide preliminary support for the judge as a measure of reference alignment, including style and content. However, this small study does not establish target-user preferences or directly compare human preferences for PAT against competing systems.

### B.4 Efficiency and Scalability

Table 12 reports training and inference times on a single NVIDIA RTX 4090 GPU. The full PAT pipeline with $T = 10$ requires approximately 35 hours of training, with much of the overhead arising from trajectory sampling. We use batched inference with vLLM to reduce this cost.

The trained modules are shared across users, so this training cost is incurred per dataset rather than per user. At inference, each request requires one generation call to each of the style agent, topic agent, and final generator. Processing the Amazon test set takes approximately 20 minutes, compared with 7 minutes for PGraph and 6 minutes for LaMP, corresponding to roughly $2.9\times$ and $3.3\times$ their inference times. PAT is faster than GraSPeR in this measured setting, but remains more expensive than the retrieval-only baselines.

Table 12: Approximate training and Amazon test-set inference times on a single NVIDIA RTX 4090 GPU. PAT trains shared modules once per dataset and requires no per-user fine-tuning.

| Method | Training | Inference |
|---|---|---|
| LaMP | N/A (zero-shot) | $\sim$6 min |
| PGraph | N/A (zero-shot) | $\sim$7 min |
| GraSPeR | $\sim$3 hrs | $\sim$3 hrs |
| PAT ($T = 10$) | $\sim$35 hrs | $\sim$20 min |

Training cost increases with the number of examples, sampled trajectories, and refinement iterations. Although we cap SFT and DPO updates within each iteration, this does not equalize total training compute across methods or iteration counts. A matched-budget comparison with simpler retrieval-plus-fine-tuning alternatives would help isolate the benefit of trajectory refinement.

## C Aggregated Results

Table 13 reports results averaged across the three benchmarks. For long-text generation, PAT achieves the highest average ROUGE-1, ROUGE-L, and METEOR scores with both backbones. Judge results are mixed: PGraph leads with Qwen3, while PAT leads with Llama3.

For short-text generation, PAT achieves the highest average ROUGE-1, ROUGE-L, and judge scores with both backbones. It also leads on METEOR with Qwen3, while PGraph leads with Llama3.

These averages complement the per-dataset results and show that the benefits vary across metrics and backbones.

Table 13: Results averaged over all benchmarks.

| Task | Metric | LLM | PAT (Ours) | GraSPeR | PGraph | LaMP |
|---|---|---|---|---|---|---|
| LONG TEXT GEN. | R-1 ↑ | *Qwen3* | **0.231** | 0.199 | 0.217 | 0.175 |
| | | *LlaMA3* | **0.235** | 0.190 | 0.208 | 0.189 |
| | R-L ↑ | *Qwen3* | **0.160** | 0.145 | 0.144 | 0.121 |
| | | *LlaMA3* | **0.164** | 0.148 | 0.147 | 0.126 |
| | MET ↑ | *Qwen3* | **0.193** | 0.158 | 0.187 | 0.144 |
| | | *LlaMA3* | **0.180** | 0.137 | 0.155 | 0.154 |
| | LLM ↑ | *Qwen3* | 3.409 | 2.917 | **3.528** | 3.064 |
| | | *LlaMA3* | **3.167** | 2.819 | 2.842 | 3.075 |
| SHORT TEXT GEN. | R-1 ↑ | *Qwen3* | **0.167** | 0.159 | 0.124 | 0.111 |
| | | *LlaMA3* | **0.147** | 0.136 | 0.128 | 0.113 |
| | R-L ↑ | *Qwen3* | **0.161** | 0.156 | 0.119 | 0.105 |
| | | *LlaMA3* | **0.144** | 0.134 | 0.122 | 0.107 |
| | MET ↑ | *Qwen3* | **0.149** | 0.126 | 0.124 | 0.113 |
| | | *LlaMA3* | 0.106 | 0.121 | **0.124** | 0.120 |
| | LLM ↑ | *Qwen3* | **3.733** | 3.321 | 3.653 | 3.443 |
| | | *LlaMA3* | **3.349** | 3.158 | 3.138 | 3.007 |

## D PAT PROMPTS

We provide the prompts used by the style agent, topic agent, and final generator.

Style Reasoning Trace Prompt $\mathcal{P}_{style}$

```
System Prompt: You are a professional writing assistant whose task
↪  is to summarize
the writing style of a user from the profile, which is past
↪  documents written by
that user. The extracted writing style summary should contain the
↪  unique features
of users writing style and preferences from the proile that are
↪  similar to the
expected output.

Your task is to summarize the user writing style from the profile
↪  in a short bullet
list (100 words or less). From the profile, you may infer the
↪  user's interests,
preference, familiarity on various topics, etc. While inferring the
↪  user's interests,
you can make reasonable guesses, e.g. people who are interested in
↪  topic A are also
likely to be interested in topic B or if they write a sentence in a
↪  specific writing
style on topic A it is likely they write it with the same style on
↪  topic B. As a
concrete example, if a user writes "I am interested in action
↪  movies" in its past
document, this is relevant to "I like to go to cinema". Another
↪  example would be if
a person prefers specific words or phrases in their writing or
↪  using a specific
grammar. You can also mention such words that they often use in
↪  your summary.

Your input:
- profile: the past documents written by the same person (might be
↪  empty).

- similar profiles: past documents written by users that have
↪  similar writing style.

Your output:
a list of bullet points and explanations describing writing style
↪  of the user.

profile:
<Profile>

similar profiles:
<Similar Profiles>

Your output should be in the following format:
Writing Style: <summarized writing style>.

DO NOT include any other text in your output.
```

Topic Reasoning Trace Prompt $\mathcal{P}_{topic}$

```
System Prompt: Your task is to summarize the product information
↪  from the product
reviews in a short bullet list (100 words or less). From the
↪  review, you may summarize
the key property of the product such as brands, makes, quality etc.
↪  While inferring
the product summary, you can make reasonable guesses, e.g. a lot of
↪  people complains
certain aspects of the products means there's certain key
↪  deficiency on the product.
You also need to summarize the ratings given to the product. You
↪  are also welcome to
make reasonable guesses, such as certain property would lead to low
↪  rating.

Your input:
- Product Reviews: the past reviews written on the product.

Your output:
a list of bullet points and explanations describing the product and
↪  the rating it
received.

profile:
<Product Text>

Your output should be in the following format:
Product Summary: <summarized product summary>.

DO NOT include any other text in your output.
```

Multi-trajectory Generation Prompt $\mathcal{P}_{gen}$

```
System Prompt: You are a personalized review generation assistant
↪  that generates
high-quality reviews based on user history and context.

Instructions: Generate a review based on the specific writing style
↪  and product
details provided below.

### Writing Style Context:
Original Example (Style Neighbor): {style_neighbor}
Style Summary: {style_summary}
### Product Context
Original Reviews (Product Neighbor): {product_neighbor}
Product Summary: {product_summary}
### Task
Review Title: {review_title}
Output Format:\nReview Text: <>
```

## E LLM-AS-A-JUDGE IN PERSONALIZATION

Prior personalization benchmarks commonly evaluate generated text using lexical overlap metrics (Au et al., 2025; Kumar et al., 2024; Salemi et al., 2024). We complement these metrics with an LLM judge that compares each generated output with its reference. We follow the prompt used in prior personalized-generation work (Salemi et al., 2025), drawing on the evaluation paradigm of Liu et al. (2023).

LLM-as-a-Judge

```
Please compare the generated text to the reference text based on
↪  how well they match
and/or are similar.

Scoring Scale:
 1. Strongly disagree
 2. Disagree
 3. Somewhat disagree
 4. Neither agree nor disagree
 5. Somewhat agree
 6. Agree
 7. Strongly agree

Content to Evaluate:
Reference Text (Ground Truth): {target_text}
Generated Text: {generated_text}

Provide only the numeric score (1-7).
```

We use Qwen2.5-7B as the judge and report scores on the seven-point scale. Because the prompt compares generated and reference texts, its score measures reference alignment rather than directly eliciting the target user's preferences. The human study in Appendix B.3 provides preliminary evidence that these scores correlate with human assessments of reference match, style consistency, and content relevance. Nevertheless, alternative outputs may reflect a user's preferences without closely matching a single reference.

## F Limitations and Further Discussion

**Compute and generalization.** The iterative training procedure requires substantial offline computation, and the current comparisons do not equalize total training budgets. Our trained models use 3B–4B backbones and English review datasets. The additional Sonnet comparison evaluates a larger model through prompting, but does not establish that the training procedure scales to larger backbones or other domains.

**Retrieval quality.** The trajectory agents depend on the relevance of retrieved writings. Missing or mismatched neighbors can introduce inaccurate style or topic evidence, particularly when the target user has little history. Training may reduce reliance on unhelpful context, but cannot guarantee recovery from retrieval errors.

**Evaluation scope.** Lexical metrics and the judge assess alignment with a reference output. The human study is small and validates the judge rather than measuring target-user satisfaction or preferences between competing systems. Broader human evaluation and repeated training runs would strengthen the evidence.

**Additional context channels.** Potential extensions include temporal evidence for changing preferences, rating and sentiment information, cohort-level signals, and cross-domain writings. These could be incorporated through additional retrieval channels and trajectory agents. Such extensions would incur additional sampling and inference costs, and their effectiveness remains to be evaluated.

**Privacy and data access.** The method conditions generation on writings from neighboring users. Deployment therefore requires appropriate authorization and access controls for retrieved content. Trajectory summaries do not provide a privacy guarantee, particularly because the generator also receives raw neighbor writings. Potential memorization or disclosure of retrieved content warrants further evaluation.

# G NOTATIONS

Table 14: Summary of key notation.

| | |
|---|---|
| $u$ | User. |
| $t$ | Topics. |
| $t_{\text{target}}$ | Target topic for personalized generation. |
| $x_{\text{target}}$ | Input corresponding to the target generation request. |
| $y$ | Ground-truth target text. |
| $\hat{y}$ | Generated personalized output text. |
| $\mathcal{H}_u$ | Historical profile of user $u$, consisting of past entries $(x_{u,n}, y_{u,n})$. |
| $(x_{u,n}, y_{u,n})$ | The $n$-th historical input-text pair for user $u$. |
| $\mathcal{A}_u$ | Auxiliary information for user $u$, defined as $(\mathcal{C}_u^{style}, \mathcal{C}_u^{topic})$. |
| $\mathcal{C}_u^{style}$ | Style context capturing stylistic cues relevant to user $u$. |
| $\mathcal{C}_u^{topic}$ | Topic knowledge context capturing topic-specific information. |
| $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ | User-topic bipartite graph. |
| $\mathcal{V}$ | Vertex set of the graph, $\mathcal{U} \cup \mathcal{T}$. |
| $\mathcal{E}$ | Edge set of the graph. |
| $\mathcal{U}$ | Set of user nodes. |
| $\mathcal{T}$ | Set of topic nodes. |
| $e_{u,t}$ | Edge between user $u$ and topic $t$. |
| $\phi(\cdot)$ | Style encoder mapping text to a continuous embedding. |
| $\mathbf{h}_u^{(0)}$ | Initial user embedding. |
| $\mathbf{h}_t^{(0)}$ | Initial topic embedding. |
| $\mathcal{R}_t$ | Set of all review texts associated with topic $t$. |
| $\mathbf{h}_v^{(k)}$ | Representation of node $v$ at GraphSAGE layer $k$. |
| $\mathbf{h}_u$ | Final user embedding after $K$ GraphSAGE layers. |
| $L$ | Number of GraphSAGE message-passing layers. |
| $\mathcal{N}_u^{style}$ | Set of stylistically similar users retrieved for user $u$. |
| $\mathcal{N}_u^{topic}$ | Set of topic-aligned neighbors retrieved for user $u$. |
| $k_1$ | Number of stylistic neighbors retained. |
| $k_2$ | Number of topic-knowledge neighbors retained. |
| $\pi_{\theta_s}$ | Style trajectory reasoning agent parameterized by $\theta_s$. |
| $\pi_{\theta_t}$ | Topic knowledge reasoning agent parameterized by $\theta_t$. |
| $\pi_{\theta_g}$ | Generation model parameterized by $\theta_g$. |
| $\mathcal{P}_{\text{style}}$ | Prompt construction function for the style reasoning agent. |
| $\mathcal{P}_{\text{topic}}$ | Prompt construction function for the topic reasoning agent. |
| $\mathcal{P}_{\text{gen}}$ | Prompt construction function for the final personalized text generation. |
| $s_u$ | Style reasoning trace produced by the style reasoning agent. |
| $p_{u,t_{\text{target}}}$ | Topic knowledge reasoning trace for user $u$ and target topic $t_{\text{target}}$. |
| $M_1$ | Number of sampled style summaries. |
| $M_2$ | Number of sampled topic summaries. |
| $s_u^{(m)}$ | The $m$-th sampled style reasoning trace. |
| $p_u^{(m)}$ | The $m$-th sampled topic reasoning trace. |
| $\hat{y}^{(m)}$ | Generated output conditioned on a sampled trajectory. |
| $R(\cdot, \cdot)$ | Downstream task reward function. |
| $R^{(m)}$ | Reward assigned to the $m$-th generated output. |
| $\theta_s, \theta_t, \theta_g$ | Parameters of the style reasoning agent, topic reasoning agent, and generator, respectively. |