PersonaX: A Recommendation Agent-Oriented User Modeling Framework for Long Behavior Sequence

Yunxiao Shi[1], Wujiang Xu[2], Zeqi Zhang[1], Xing Zi[1], Qiang Wu[1], Min Xu[1]
[1]University of Technology Sydney, [2]Rutgers University
Yunxiao.Shi@student.uts.edu.au, Min.Xu@uts.edu.au

Abstract

User profile embedded in the prompt template of personalized recommendation agents play a crucial role in shaping their decision-making process. High-quality user profiles are essential for aligning agent behavior with real user interests. Typically, these profiles are constructed by leveraging LLMs for user profile modeling (LLM-UM). However, this process faces several challenges: (1) LLMs struggle with long user behaviors due to context length limitations and performance degradation. (2) Existing methods often extract only partial segments from full historical behavior sequence, inevitably discarding diverse user interests embedded in the omitted content, leading to incomplete modeling and suboptimal profiling. (3) User profiling is often tightly coupled with the inference context, requiring online processing, which introduces significant latency overhead. **In this paper, we propose PersonaX, an agent-agnostic LLM-UM framework to address these challenges. It augments downstream recommendation agents to achieve better recommendation performance and inference efficiency.** PersonaX (a) segments complete historical behaviors into clustered groups, (b) selects multiple sub-behavior sequences (SBS) with a balance of prototypicality and diversity to form a high-quality core set, (c) performs offline multi-persona profiling to capture diverse user interests and generate fine-grained, cached textual personas, and (d) decouples user profiling from online inference, enabling profile retrieval instead of real-time generation. **Extensive experiments demonstrate its effectiveness: using only 30–50% of behavioral data (sequence length 480), PersonaX enhances AgentCF by 3–11% and Agent4Rec by 10–50%.** As a scalable and model-agnostic LLM-UM solution, PersonaX sets a new benchmark in scalable user modeling. The code is available at URL [1].

[1]https://github.com/Ancientshi/PersonaX

1 Introduction

Recent advances in LLMs (Yang et al., 2023; Qin et al., 2023; Xu et al., 2025a,b) have enabled instruction-based agents (Xu et al., 2025b) to excel in autonomous interaction and decision-making (Li et al., 2023; Wang et al., 2024a,c). By integrating realistic user profiles into prompts, these agents achieve personalization and effectively mimick real user behaviors (Sun et al., 2024; Shao et al., 2023). Personalized recommendation agents—such as AgentCF (Zhang et al., 2024b), Agent4Rec (Zhang et al., 2024a), and RecAgent (Wang et al., 2024b)—inherit this potential yet face a challenge: users seldom state their preferences explicitly, leaving them implicit in their historical behavioural traces. Hence, modeling representative descriptive user profiles from implicit feedback becomes crucial for unleashing the full power of personalized recommendation agents.

Recommendation agents typically employ large language models for real-time user modelling (LLM-UM). The profile produced by the LLM is embedded in the prompt and guides the model when generating recommendations for a target item. Existing LLM-UM methods can be grouped into three categories. **Demonstration** approaches encode the user's behavior sequence (BS) directly into the prompt as in-context examples, allowing the LLM to generalize from explicit demonstrations (Pi et al., 2020; Dai et al., 2023; Liu et al., 2023). **Summarization** techniques distill extensive interaction histories into concise textual personas that capture core preference signals; this strategy, shown to improve personalization (Richardson et al., 2023), is adopted by ONCE (Liu et al., 2024) and Agent4Rec (Zhang et al., 2024a). Moreover, methods like AgentCF (Zhang et al., 2024b) and RecAgent (Wang et al., 2024b) adopt a **Reflection** approach, employing reflection mechanisms (Cheng et al., 2023; Zhao et al., 2024a; Shi et al.,

Findings of the Association for Computational Linguistics: ACL 2025, pages 5764–5787
July 27 - August 1, 2025

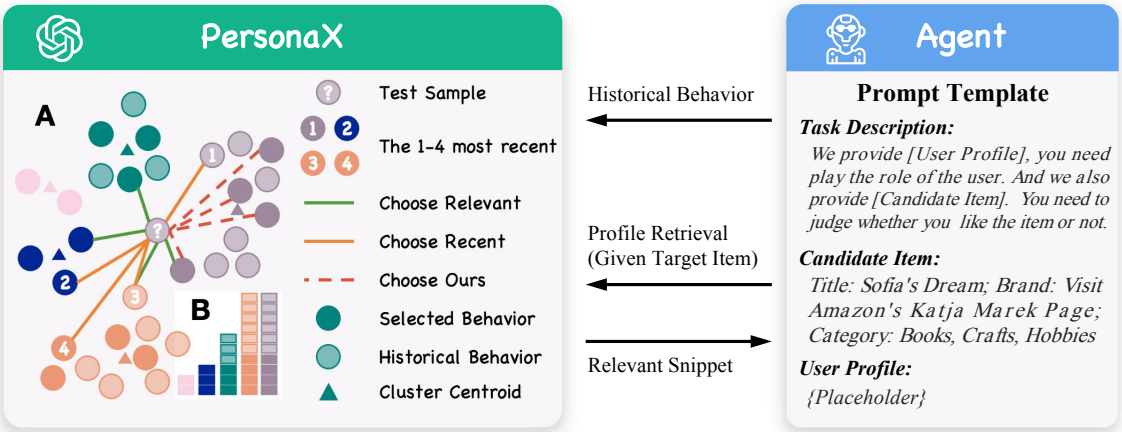

Figure 1: PersonaX is a user-modeling tool that leverages historical behavioral data to construct multiple persona, which is cached for retrieval by downstream agent. In the left panel, PersonaX.A visualizes the overall behavior distribution, the results of clustering, and selected/unselected samples. PersonaX.B, depicts how sampling budgets are allocated at a 50% selection ratio.

2024a) on the behavior sequence to iteratively refine user persona.

LLM-UM depends on historical behavioral sequences (BS) that encode rich preference signals. A straightforward strategy is to employ the **Full data** into the LLM for profiling (Zhang et al., 2024a), but this quickly becomes infeasible as sequence length grows. **Recent sampling** truncates the user's behavior sequence to its most recent interactions, prioritizing short-term interests (Hou et al., 2024; Zhang et al., 2024b). This strategy depends on temporal information that are only available during online inference. Consequently, both the sampling and the subsequent LLM-UM process must be performed online. Alternatively, **Relevance sampling** (Salemi et al., 2024; Zhang et al., 2024a; Zhou et al., 2024) selects those past behaviors most pertinent with the target item for capturing long-term preference patterns. Like Recent sampling, it relies on item-specific contextual signals that only become available at inference time—so this strategy also must be executed online.

We identify three principal limitations in the way current LLM-UM methods utilize historical behavioral data: (1) **Difficulty profiling from long behavior sequences.** Summarization-based approaches are constrained by LLMs' maximum input length and suffer from the "lost-in-the-middle" phenomenon (Zhao et al., 2024b; Shi et al., 2024b,c; Borgeaud et al., 2022; Lewis et al., 2020), whereby critical mid-sequence context is omitted, undermining accurate preference inference. Reflection-based methods, in turn, incur prohibitive computational cost and latency when reasoning over very lengthy sequences. Furthermore, excessive behavioral data introduce noise and redundancy that obscure truly salient signals. (2) **Sampling inevitably incurs information loss.** By omitting valuable behavioral signals, existing sampling strategies can compromise the quality of the generated user profile. (3) **Profiling relies on online contextual information and results in inference latency.** Both recent and relevance sampling strategies require real-time contextual inputs (e.g., current timestamp, target item), which mandates modeling user profiles at inference time and incurs decision-making latency overhead.

To address these challenges, we propose **PersonaX**, a novel LLM-UM framework that performs end-to-end profile from long behavioral sequences in an offline setting, cached for online retrieval by downstream recommendation agents for decision making. Such paradigm significantly reduces online inference latency and enhances overall recommendation performance, their architecture is illustrated in Figure 1. PersonaX segments full historical behaviors into clustered groups and selects sub-behavior sequences (SBS) for each cluster with prototypicality-diversity balanced sampling. Summarization or Reflection is conducted on SBS in an offline manner, generating multiple fine-grained personas that capture diverse user interests. Those cached textual representations are then retrieved by downstream recommendation agents during online inference stage. PersonaX prioritizes example quality over quantity, using only a small fraction of behavioral data (sequence length < 5) to select compact yet informative SBS and avoid selecting irrelevant or noisy samples, thus overcoming issues such as input length constraints and mid-content

oversight. Compared to both Recent and Relevance sampling, our method constructs a high-fidelity core-set that preserves the full spectrum of user interests, thereby avoiding information loss. PersonaX assumes responsibility for profile modeling, enabling cached profile retrieval instead of on-the-fly generation and thereby eliminating application agents' online inference latency.

In summary, our contributions are threefold. (1) **PersonaX Framework**: We introduce PersonaX, an LLM-based user-profile modeling framework oriented for recommendation agents. By decoupling profile generation from online inference, PersonaX eliminates real-time modeling overhead—accelerating inference—and delivers more representative user profiles that substantially boost downstream recommendation performance. (2) **Data-Efficient Core Behavior Selection**: We introduce a novel strategy for selecting core behaviors via clustering, adaptive allocation of sampling budgets, and a prototypicality-diversity-balanced in-cluster selection mechanism. By using only 30–50% of the data utilization and ignore other redundant behaviors, our method generates multiple compact sub-behavior sequences (SBSs), each capturing a distinct facet of user preferences. (3) **Extensive Validation**: We evaluate PersonaX with two leading recommendation agents—AgentCF and Agent4Rec— on next-item ranking tasks across three datasets of varying sequence lengths. PersonaX consistently boosts ranking accuracy (3–11% for AgentCF; 10–50% for Agent4Rec) and accelerates online inference efficiency.

2 Preliminary

2.1 User Modeling

Let $\mathcal{S} = \{(I_1, L_1), (I_2, L_2), \ldots, (I_n, L_n)\}$ denotes a user's historical behavior sequence of length n, where I_i represents the i-th interacted item and $L_i \in \{0, 1\}$ indicates the corresponding interaction label (0 for dislike and 1 for like). We define the task of user modeling is to construct a precise and representative user persona $\mathcal{P}(\mathcal{S})$ by leveraging the historical behavioral data $\mathcal{S}$, where $\mathcal{P}(\cdot)$ is a user modeling method (e.g., Summarization and Reflection). The learned user persona should capture the implicit preference patterns underlying interactions, enabling augmentation for downstream instructional agent recommendation.

2.2 Sub-Behavior Sequence (SBS) Selection.

To tackle the challenge of LLM-UM struggling with analyzing long behavior sequence, sampling methods are often employed on the full historical sequence $\mathcal{S}$. These methods aim to extract a Sub-Behavior Sequence (SBS) that retains the most essential information necessary for accurate user profiling while significantly reducing sequence length. Formally, let $\mathcal{S}^* = \{\hat{I}_1, \hat{I}_2, \ldots, \hat{I}_k\} \subseteq \mathcal{S}$ denote the SBS of length k ($k \ll n$), where $\hat{I}_i$ represents the i-th selected behavior. The selection ratio, $\frac{k}{n}$, quantifies the compression achieved.

3 Method

3.1 Behavior Clustering

We employ hierarchical clustering to group items based on user interest similarity, treating each cluster as a cohesive analysis unit. A language embedding model $\mathbf{E}(\cdot)$, such as BGE Embedding (Chen et al., 2024) or EasyRec (Ren and Huang, 2024), encodes each item I_i into a dense vector $\mathbf{e}_i$. Let $\mathcal{E} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n\}$ represent the item embeddings from the user's interaction history. Pairwise similarity is measured via Euclidean distance: $d(\mathbf{e}_i, \mathbf{e}_j) = \|\mathbf{e}_i - \mathbf{e}_j\|_2$, denoted as $d_{i,j}$.

Clustering is controlled by a distance threshold τ, which restricts the maximum intra-cluster distance while preventing merges between clusters with inter-cluster distances below τ. The resulting clusters $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$ satisfy Intra-cluster constraint: $\forall c \in \mathcal{C}, \forall I_i, I_j \in c, d_{i,j} < \tau$ and Inter-cluster constraint: $\forall c_p, c_q \in \mathcal{C}, c_p \neq c_q, d(c_p, c_q) \geq \tau$.

3.2 Sampling Budget Allocation

Given a finite budget k for sampling historical behaviors, we propose a Sampling Budget Allocation Strategy to distribute this budget across clusters. The algorithm dynamically adjusts allocation based on cluster size distribution, ensuring that smaller clusters are given sufficient attention while preventing larger clusters from dominating the selection process. This promotes a balanced distribution of selected samples, preserving the diversity of sampled behaviors and maintaining a representative coverage of the data (Zheng et al., 2023).

The strategy first sorts clusters by size in ascending order. Each cluster is initially assigned an average allocation q. If a cluster's size is smaller than q, it receives its exact size, and q is recalculated based on the remaining quota. Otherwise, the cluster is

Algorithm 1 Sampling Budget Allocation

Input: Set of clusters $\mathbf{C} = \{c_1, c_2, \ldots, c_m\}$, Cluster size list $\mathbf{s} = \{s_1, s_2, \ldots, s_m\}$ where $s_i = |c_i|$, Total sampling budget k
Output: Allocation list $\mathbf{A} = \{a_1, a_2, \ldots, a_m\}$
function AllocateBudget($\mathbf{C}, \mathbf{s}, k$)
  Sort $\mathbf{s}$ in ascending order and obtain sorted indices $\mathbf{I}$
  Initialize allocation $\mathbf{A} \leftarrow [0, 0, \ldots, 0]$
  Remaining budget $B \leftarrow k$
  for each cluster i in sorted order **do**
    $r \leftarrow$ number of remaining clusters
    $q \leftarrow B \,//\, r$ ▷ Average allocation per remaining cluster
    $a_i \leftarrow \min(s_i, q)$ ▷ Allocate min of cluster size or q
    $B \leftarrow B - a_i$ ▷ Update remaining budget
  end for
  while $B > 0$ **do** ▷ Distribute any remaining budget
    for each cluster i **in sorted order if** $a_i < s_i$ **do**
      $a_i \leftarrow a_i + 1$
      $B \leftarrow B - 1$
      if $B = 0$ **then break**
      end if
    end for
  end while
  Restore original order for $\mathbf{A}$ using $\mathbf{I}$
  return $\mathbf{A}$
end function

allocated q. This process repeats iteratively until the entire budget is assigned. Algorithm 1 details the method, and Figure 1.B illustrates an example, where smaller clusters are fully allocated first, and the remaining budget is evenly distributed among larger clusters.

3.3 In-Cluster Selection

After partitioning user behaviors into semantically coherent clusters and each cluster is allocated with a sampling quota, we are to select a representative subset from each cluster. Data selection methods that greedily choose items closest to the cluster centroid (e.g., (Welling, 2009; Rebuffi et al., 2017; Sorscher et al., 2022)) yield overly homogeneous user profiling, while boundary-focused strategies (e.g., (Paul et al., 2021; Toneva et al., 2019)) risk overemphasizing diversity at the expense of prototypical patterns. To address these issues, we introduce a sampling strategy that balances prototypicality and diversity within each cluster. For a cluster c_i, its centroid is computed as $\mu_i = \frac{1}{|c_i|} \sum_{I_j \in c_i} \mathbf{E}(I_j)$. Let c_i^* denote the selected subset from c_i. Our goal is to maximize both the similarity of selected items to the centroid and the diversity among them:

Algorithm 2 In-Cluster Selection

Input: Cluster $c_i = \{I_1, I_2, \ldots, I_{n_i}\}$, centroid μ_i, selection size a_i, weights w_p and w_d.
Output: Sub-Behavior Sequence S_i^*.
function DynamicSelect($c_i, \mu_i, a_i, w_p, w_d$)
  Initialize $c_i^* \leftarrow \emptyset$
  Compute item embeddings $\mathbf{E}(c_i) = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{n_i}\}$, where $\mathbf{e}_j$ is the embedding of item $I_j \in c_i$.
  Select the initial item:

$$\mathbf{e}_{\text{init}} = \arg \min_{\mathbf{e}_j \in \mathbf{E}(c_i)} d(\mathbf{e}_j, \mu_i)$$

  Update $c_i^* \leftarrow c_i^* \cup \{\mathbf{e}_{\text{init}}\}$ and $\mathbf{E}(c_i) \leftarrow \mathbf{E}(c_i) \setminus \{\mathbf{e}_{\text{init}}\}$.
  while $|c_i^*| < a_i$ **do**
    Compute Marginal Gains:
    for all $\mathbf{e}_j \in \mathbf{E}(c_i)$ **do**
      Compute prototypicality gain:

$$g_p(\mathbf{e}_j) = \frac{w_p}{1 + d(\mathbf{e}_j, \mu_i)}$$

      Compute diversity gain:

$$g_d(\mathbf{e}_j) = \frac{2w_d}{c_i} \sum_{\mathbf{e}_b \in c_i^*} d(\mathbf{e}_j, \mathbf{e}_b)$$

    end for
    Evaluate Selection Priority:
    Identify the item maximizing the combined gain:

$$\mathbf{e}_j^* = \arg \max_{\mathbf{e}_j \in \mathbf{E}(c_i)} (g_p(\mathbf{e}_j) + g_d(\mathbf{e}_j))$$

    Update $c_i^* \leftarrow c_i^* \cup \{\mathbf{e}_j^*\}$ and $\mathbf{E}(c_i) \leftarrow \mathbf{E}(c_i) \setminus \{\mathbf{e}_j^*\}$.
  end while
  Chronologically sort c_i^* to get S_i^*.
  return S_i^*
end function

$$\max_{c_i^*} \left(w_p \cdot \sum_{I_j \in c_i^*} \frac{1}{1 + d(\mathbf{e}_j, \mu_i)} + w_d \cdot \frac{2}{a_i} \sum_{\substack{I_a, I_b \in c_i^* \\ a \neq b}} d(\mathbf{e}_a, \mathbf{e}_b) \right)$$

Here, $w_p = \alpha^{-10}$ and $w_d = 1 - w_p$, with the hyperparameter α tuning the trade-off: values near 1.001 approximate centroid selection, while values around 1.4 approach boundary selection. Empirically, α is set between 1.06 and 1.08 (see Section 5.4). We frame the selection as discrete optimization problem and using a Greedy Selection algorithm (Algorithm 2) to solve it, which iteratively selects the element with the highest marginal gain. A visual explanation of the selection algorithm is provided in Appendix E, and Appendix F provides a convergence analysis of the proposed objective function and demonstrates how the greedy algorithm can attain suboptimal performance.

Table 1: Time complexity analysis. Cluster, A.1 and A.2 refers to clustering method used in Section 3.1, Algorithms 1 and 2, respectively.

Agent LLM-UM Strategy	Offline Phase Complexity	Online Phase Complexity
AgentCF$_{\text{Recent + Reflection}}$	–	$O(2kT + N_I \cdot T)$
AgentCF$_{\text{Relevance + Reflection}}$	$O(nd)$	$N_I \cdot O(2kT + d + T)$
Agent4Rec$_{\text{Recent + Summarization}}$	–	$O(T + N_I \cdot T)$
Agent4Rec$_{\text{Relevance + Summarization}}$	$O(nd)$	$N_I \cdot O(d + 2T)$
AgentCF$_{\text{PersonaX}}$	$O(C \cdot 2kT + nd + \text{Cluster} + \text{A.1} + \text{A.2})$	$N_I \cdot O(T + d)$
Agent4Rec$_{\text{PersonaX}}$	$O(CT + nd + \text{Cluster} + \text{A.1} + \text{A.2})$	$N_I \cdot O(T + d)$

The design of our objective function is inspired by prior studies in data selection and empirical findings (Sorscher et al., 2022), which demonstrate that for small datasets, prioritizing simple, prototypical examples yields the greatest benefit, whereas for sufficiently large datasets, selection methods that emphasize harder examples improve the generalization of deep learning models.

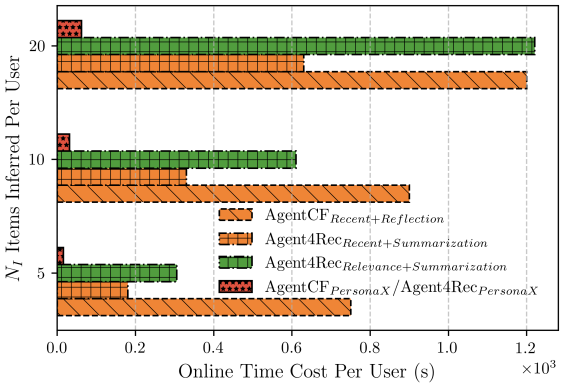

Figure 2: Online time cost analysis.

3.4 Offline Profiling and Online Selection

After selecting representative SBS, PersonaX continue to construct persona offline. Given a selected behavior subset c_i^*, we generate a corresponding persona $p_i = \mathcal{P}(c_i^*)$. To ensure contextually relevant recommendations, PersonaX retrieves the most pertinent persona snippet P_{selected} online, which is integrated into prompt templates to instruct agent recommendation.

4 Efficiency Analysis

Recent sampling operates in constant time $O(1)$. Let $O(d)$ denotes the time required to encode an item into an embedding vector, and $O(n \log k)$ represents the complexity of selecting the k most relevant/recent items from n items. Thus Relevance sampling has a time complexity of $O(nd + n \log k)$. For SBS sampling applied in PersonaX, we use $O(\text{Cluster} + \text{Alg.1} + \text{Alg.2})$ represent the time cost for process we depict from Section 3.1 to 3.3. Let C denote the number of clusters, $O(T)$ the time complexity of a single API request to the LLM, and N_I the number of candidate items inferred per time. We perform an analysis of the time complexity during both offline and online stages associated with two recommendation agents AgentCF and Agent4Rec. Additionally, we evaluate their time cost with the use of PersonaX for comparison. The results are summarized in Table 1. The detailed illustrations are provided in Appendix A.

The primary contributors to time consumption are T and d, while $O(C)$, $O(n \log k)$, $O(\text{Cluster+A.1+A.2})$ in ranking, clustering, and sampling are negligible. Assuming $n = 500$, $C = 20$, $T = 3$, $d = 0.1$, and $k = 10$, and varying N_I over 5, 10, and 20. Additionally, to model a realistic production setting in which recommendation agents server online continuously, we assume that persona in PersonaX is cached for $\mathcal{D} = 10$ successive inference calls. In practice, this conservative threshold means the persona is refreshed whenever ten new user behaviors are observed, preventing profile staleness; the actual refresh interval can be tuned empirically. Because vanilla AgentCF and Agent4Rec must regenerate the user profile at every call, their online latency is multiplied by $\mathcal{D}$, whereas PersonaX-assisted variants avoid this overhead altogether. Figure 2 visualises the resulting online time consumption. The bar for AgentCF$_{\text{Relevance+Reflection}}$ is omitted because its latency is orders of magnitude higher than that of the other methods and would dominate the plot. We mainly make a comparison when backbone recommendation agent as Agent4Rec which is more realistic, we observe that PersonaX-assisted Agent4Rec reduces runtime by 95% compared with the Agent4Rec variant that employs Relevance sampling, while recudes runtime by 91% for variants that use Recent sampling.

5 Experiments

In this section, we are to address these research questions (**RQs**): • **RQ1**: How does PersonaX improve downstream agent recommendation, and how the performance compared with baseline approaches? • **RQ2**: How does the sampling size of historical behaviors affect the efficacy of user modeling? • **RQ3:** How sensitive is our method to hyper-parameter settings, and how can optimal parameters be chosen?

5.1 Experimental Setup

5.1.1 Datasets

We evaluate on two widely used subsets of the Amazon review dataset (Ni et al., 2019): *CDs and Vinyl* and *Books*. For the CDs dataset, similar to the settings in (Zhang et al., 2024b), we consider two variants, $\texttt{CDs}_{50}$ and $\texttt{CDs}_{200}$, which have average user interaction sequence lengths of 50 and 200, respectively. For the Books dataset, rather than restricting each user's interactions to 20 items as in (Zhang et al., 2024a), we adopt the approach outlined in (Pi et al., 2019, 2020) to construct longer sequences, resulting in $\texttt{Books}_{480}$. A more detailed description, statistical analysis, and reproducibility are provided in Appendix B.

5.1.2 Evaluation

We utilize all the interaction data except the most recent one to construct the user's behavior history (Kang and McAuley, 2018). And the most recent interaction is reserved for testing. We randomly sample 9 negative items and combine them with the positive item, converting these 10 items into textual descriptions to form the candidate set. For evaluation metric, we adopt the typical top-N metrics hit rate (HR@{1, 5}), normalized discounted cumulative gain (NDCG@{5}) (Järvelin and Kekäläinen, 2002) and Mean Reciprocal Rank (MRR@{10}) (Sarwar et al., 2001). For all evaluation metrics in our experiments, higher values indicate better performance. Also, an intuitive case study is provided in Appendix G.

5.1.3 Downstream Recommendation Agent

We select two recommendation agents **AgentCF** (Zhang et al., 2024b) which models user personas using a Reflection mechanism, and **Agent4Rec** (Zhang et al., 2024a) which captures users' unique preferences through a Summarization method. Further details on the foundational methods can be found in Appendix C. The original AgentCF offers two configurations—AgentCF (Recent+Reflection) and AgentCF (Relevance+Reflection)—while the standard Agent4Rec corresponds to Agent4Rec (Full+Summarization).

5.1.4 Baseline Comparison

To rigorously assess the benefit of integrating PersonaX, we enlarge the comparison scope beyond a mere juxtaposition of PersonaX-assisted AgentCF and Agent4Rec with their original implementations. Specifically, we pair two LLM-UM methods—Reflection and Summarization—with six representative behavior-sequence sampling strategies that serve as baselines: (1) Full (Zhang et al., 2024a): Using complete user behavior sequence. (2) Recent (Zhang et al., 2024b): Selecting the most recent behaviors to capture the user's short-term preferences. (3) Relevance (Zhang et al., 2024b; Pi et al., 2020): Retrieving the subset of behaviors most pertinent to the recommendation scenario from the user's long-term preferences. (4) Random (Guo et al., 2022; Prabhu et al., 2020): Randomly selecting a portion of behaviors, it is a robust and effective sampling method. (5) Centroid Selection (Welling, 2009; Rebuffi et al., 2017; Sorscher et al., 2022): As outlined in Section 3.3, we configure $\alpha = 1.001$ in Algorithm 2. This configuration prioritizes the selection of samples that are closest to the cluster centroid, effectively capturing the most prototypical data points within the cluster. (6) Boundary Selection (Paul et al., 2021; Toneva et al., 2019): As detailed in Section 3.3, we set $\alpha = 1.4$ in Algorithm 2. Under this setting, the algorithm selects samples located at the cluster boundary and emphasizes the diversity coverage.

5.1.5 Implementation Details

We applied AgentCF to $\texttt{CDs}_{50}$, and Agent4Rec for $\texttt{CDs}_{200}$ and $\texttt{Books}_{480}$. For PersonaX, extensive experiments were conducted under diverse hyper-parameter configurations: the distance threshold $\tau \in \{0.5, 0.7\}$ and the trade-off parameter $\alpha \in \{1.01, 1.04, 1.08, 1.1\}$. Different selection ratios ($\frac{k}{n}$) were tested, including $\{10, 30, 50, 70, 90, 100\}$ for all three datasets. We also ensured that each cluster sampled at least one behavior by enforcing $k = \min(m, n \cdot \text{ratio})$. To evaluate the performance of the baseline methods, we varied the hyper-parameter selection ratio across different ranges for each dataset. Specifically, for $\texttt{CDs}_{50}$, the selection ratio was chosen from $\{0.02, 0.06, 0.08, 0.1, 0.16, 0.2, 0.3\}$.

Table 2: Performance comparison study.

LLM-UM	Reflection				Summarization							
Datasets	CDs_{50}				CDs_{200}				Books_{480}			
Metrics	Hit@1	Hit@5	NDCG@5	MRR@10	Hit@1	Hit@5	NDCG@5	MRR@10	Hit@1	Hit@5	NDCG@5	MRR@10
Full	19.00	66.00	42.56	39.38	36.00	67.00	51.75	50.78	19.00	50.00	34.59	35.76
Random	31.00	67.00	49.18	47.74	36.00	68.00	51.26	50.24	33.00	73.00	53.59	50.50
Recent	34.00	69.00	50.69	49.31	39.00	68.00	53.89	53.34	35.00	74.00	55.23	52.76
Relevance	40.00	69.00	54.97	54.47	51.00	73.00	61.73	61.98	61.00	80.00	71.50	71.86
Centroid	43.00	66.00	55.21	55.91	42.00	70.00	57.07	56.53	60.00	81.00	71.61	70.67
Boundary	42.00	68.00	55.85	55.73	48.00	66.00	57.13	58.71	58.00	80.00	70.38	69.55
Ours	**45.00**	**72.00**	**57.34**	**58.38**	**55.00**	**75.00**	**64.56**	**65.06**	**65.00**	**83.00**	**74.26**	**73.22**

Similarly, for CDs_{200}, it ranged over $\{0.005, 0.01, 0.02, 0.03, 0.05, 0.08, 0.1\}$, and for Books_{480}, the selection ratio spanned $\{0.002, 0.005, 0.008, 0.011, 0.014\}$. These selection ratios settings were made to evaluate the baseline methods at equivalent levels of data resource utility, ensuring a fair and controlled comparison with PersonaX whose SBS sizes are listed in Table 3. The prompt templates are provided in Appendix H.

5.2 Performance Evaluation (RQ 1)

Key observations and insights from Tables 3 highlight the robustness and effectiveness of our proposed method across various agent recommendation approaches, datasets, and evaluation metrics. PersonaX consistently outperforms the Full approach under any level of data resource utilization, even in scenarios where PersonaX achieves its least favorable results. Notably, on the Books_{480} dataset, which features longer behavior sequences, our method achieves significant improvements over the Full methods. This phenomenon highlights the shortcomings of existing agent recommendation methods in handling long behavior sequences, but PersonaX fills this critical research gap.

Table 2 reports the best MRR@10, highlighting PersonaX's performance advantages over baselines. Our approach demonstrates substantial improvements over the widely adopted and strong baseline method, Relevance. For example, on the CDs_{50} dataset, our method achieves a Hit@1 score of 45.00, significantly exceeding the 40.00 obtained by Relevance. Similarly, we observe the suboptimal performance of the Centroid and Boundary methods, particularly on CDs_{200}. Upon analysis, we attribute the underperformance of the Centroid method to its tendency to sample overly homogeneous information, which results in overly simplistic and narrow user personas. While the Boundary method ensures sample diversity, an excessive focus on diversity can dilute the representation of typical user persona characteristics. In contrast, our method consistently delivers superior and stable performance, highlighting the effectiveness of balancing prototypicality and diversity. This equilibrium enables our approach to capture nuanced user personas with greater precision, establishing it as a robust and versatile solution for user modeling.

Table 3: Performance of PersonaX at different selection ratios. We highlight **best performance**, and the **worst performance**.

Reflection on CDs_{50}

Ratio	#SBS	HR@1	HR@5	NDCG@5	MRR
100	5.56	41.00	67.00	**54.67**	54.67
90	4.69	42.00	69.00	55.66	55.22
70	3.52	**39.00**	70.00	54.95	**53.50**
50	2.88	41.00	67.00	54.69	55.08
30	1.83	**45.00**	**72.00**	**57.34**	**58.38**
10	1.0	42.00	**66.00**	56.07	55.25

Summarization on CDs_{200}

Ratio	#SBS	HR@1	HR@5	NDCG@5	MRR
100	8.15	**43.00**	**68.00**	**56.85**	**57.07**
90	7.19	49.00	70.00	59.66	59.95
70	5.48	47.00	71.00	60.54	60.54
50	3.59	**55.00**	**75.00**	**64.56**	**65.06**
30	2.3	51.00	73.00	62.45	62.42
10	1.0	47.00	72.00	61.91	60.99

Summarization on Books_{480}

Ratio	#SBS	HR@1	HR@5	NDCG@5	MRR
100	15.35	61.00	83.00	73.56	72.18
90	11.74	**59.00**	**80.00**	**71.36**	**71.70**
70	8.41	64.00	81.00	72.55	72.62
50	4.2	**65.00**	**83.00**	**74.26**	**73.22**
30	1.82	64.00	82.00	73.68	72.14
10	1.0	63.00	83.00	72.90	71.75

5.3 Sampling Size Investigation (RQ 2)

Understanding the influence of sequence length of SBS on the efficacy of user modeling is a pivotal research question. Traditional recommendation systems have largely relied on long-sequence modeling strategies, such as SIM (Pi et al., 2020), which, when applied to datasets like Amazon Books, typically sample 10 interactions to approximate short-term behavioral patterns and 90 interactions for long-term modeling. However, in the context of LLM-UM, prior works such as AgentCF and Agent4Rec have yet to conduct a systematic investigation into the effect of sequence length on user

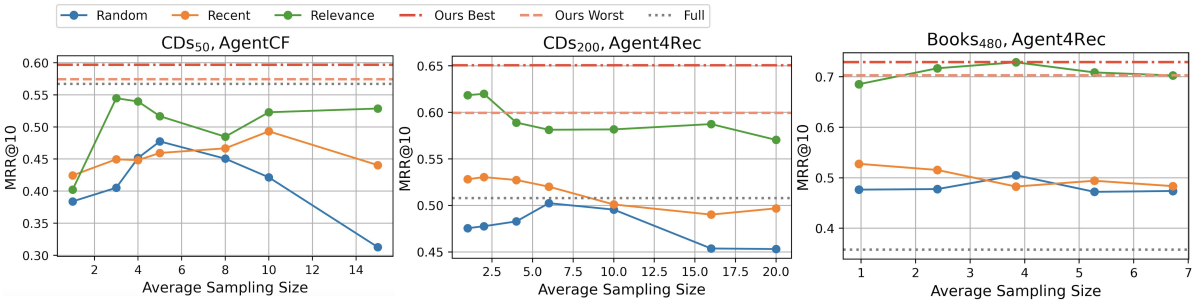

Figure 3: Analysis of the impact of sampling size on user modeling.

modeling performance.

To address this gap, we first conduct analysis on PersonaX. As shown in Tables 3, the results indicate that performance generally peaks at intermediate selection ratios or short SBS lengths. For instance, 30% selection ratio for $\texttt{CDs}_{50}$ and 50% for both $\texttt{CDs}_{200}$ and $\texttt{Books}_{480}$. We further examined the performance of three sampling strategies—Random, Recent, and Relevance—under varying sampling sizes, as illustrated in Figure 3, finding that while initial increases in sampling size improve performance, oversampling eventually leads to performance deterioration. The optimal sampling size varies across datasets. Specifically, for the Relevance method, the ideal size is approximately 3, while the Recent method demonstrates heightened sensitivity to dataset characteristics, with the most recent single behavior often yielding strong results. For the Random method, a sampling size of around 5 is most effective.

5.4 Hyper-parameter Analysis (RQ3)

This section examines the impact of τ and α on PersonaX's performance. Our emperimental results, as illustrated in Figure 4, uncovers nuanced patterns in how these hyperparameters influence the model's overall performance. In Appendix D, we present more illustration alongside a visualization analysis of the sampling process.

The empirical results indicate that: (1) incorporating diverse samples is beneficial for enhancing performance. Specifically, higher values of α (e.g., 1.06–1.08) lead to significant performance improvements at large ratios (0.5–0.9); (2) PersonaX requires minimal fine-tuning effort within the range $\tau \in [0.5, 0.7], \alpha \in [1.04, 1.08]$ and demonstrates robust performance, with a worst-case accuracy of 71.6, which closely approaches the best performance of the relevance baseline (71.86); (3) a higher τ expands the behavioral scope within clusters, making a lower α preferable to prevent excessive diversity. Conversely, a larger α prioritizes more diverse samples, necessitating a smaller τ to mitigate over-dispersion. For instance, when $\tau = 0.5$, a higher α (1.08) is appropriate, whereas $\tau = 0.7$ favors a slightly lower α (1.06) to avoid overemphasizing highly diverse samples.

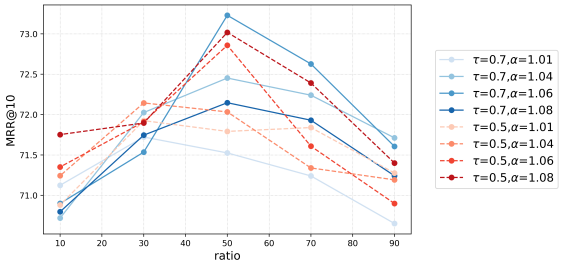

Figure 4: Impact of τ and α on PersonaX.

6 Related Works

6.1 Large Language Model for User Modeling

User Modeling (UM) aims to extract valuable insights and patterns from users' long historical behavior sequences, and Large Language Models (LLMs) excel in characterizing user personalities and discerning preferences. Leveraging LLMs for UM has gained increasing attention, and the generated textual personas can be applied to downstream personalization tasks (Xu et al., 2024a; Mei and Zhang, 2023; Xu et al., 2023, 2024b). For example, ONCE (Liu et al., 2024) utilizes ChatGPT to infer users' preferred topics and regions, enhancing click-through rate prediction with these generated profiles. Kang et al. (Kang et al., 2023) enable LLMs to comprehend user preferences from behavior history to predict user ratings. LLMRec (Lyu et al., 2024) identifies limitations in directly using raw item descriptions, which often fail to capture the subtle nuances of user preferences. To address this, it employs four distinct text enrichment strategies to enhance the input and improve recommendation performance. LLMRank (Hou et al., 2024) introduces specialized prompting and bootstrapping techniques that incorporate user interaction histories, effectively aligning with user intent. More-

over, two prominent strategies—Summarization and Reflection—have been widely adopted in leading agent recommendation frameworks, such as Agent4Rec (Zhang et al., 2024a), RecAgent (Wang et al., 2024b), and AgentCF (Zhang et al., 2024b). Summarization focuses on distilling user behaviors, while reflection emphasizes iterative learning from interactions.

However, no research has focused on the performance of LLMs when handling extensive behaviors, nor has any LLM-UM method been proficient at efficiently and accurately modeling user personas from long behavior sequences. We are the first to address this gap and introduce PersonaX.

6.2 Personalized Agents

LLM-driven agents have gained prominence for their autonomous decision-making, tool utilization (Yang et al., 2023; Qin et al., 2023; Xu et al., 2025a,b; Mei et al., 2024), and adaptive intelligence. Recent advances enable personalized agents through encoded personalities (Rao et al., 2023), backgrounds, and behavioral traits in prompts. Such persona-driven frameworks enhance user engagement through human-like interactions (Sun et al., 2024), with applications like CharacterAgent (Shao et al., 2023) demonstrating consistent persona emulation of historical figures for immersive simulations. The personalization of agent also enable the simulations of social dynamics (Park et al., 2023), competition (Zhao et al.), and collaboration (Tran et al., 2025).

However, recommendation agents (Zhu et al., 2025) face distinct challenges: Unlike predefined personas, user preferences in recommendation contexts are implicit and behaviorally embedded rather than verbally expressed. This creates alignment difficulties between agent decisions and users' latent preferences. The primary objective of PersonaX is to develop a highly accurate and realistic user modeling method, enabling instruction-based agents to consistently simulate and align with the decision-making behaviors of the users they surrogate.

7 Conclusion

In this study, we present PersonaX, a LLM-UM framework oriented for agent recommendation specially designed for processing long behavior sequences. PersonaX utilizes only 30%-50% of the historical behavior data and strategically select high-quality sub-behavior sequences of short length (often < 5) for generating broad spectrum of persona snippets offline. When PersonaX integrated into existing agent recommendation methods, such as AgentCF and Agent4Rec, PersonaX delivers substantial performance gains—ranging from 3% to 11% over AgentCF, and an impressive 10% to 50% improvement over Agent4Rec. Theoretical analysis indicates that integrating PERSONAX into downstream recommendation agents markedly reduces online inference latency—a benefit that is especially pronounced in continuously servers. We believe that PersonaX significantly facilitate the agent recommendation in predictive accuracy and inference efficiency.

Limitations

While PersonaX effectively tackles the challenge of modeling user behavior over extended sequences in LLM-based user modeling, its performance in real-world streaming data scenarios remains unexplored. This presents a promising opportunity for future enhancements. A fundamental characteristic of PersonaX lies in its offline generation of multiple personas, capturing diverse aspects of user preferences. This design facilitates long-horizon modeling, where personas encapsulate user interests over extended periods and maintain their effectiveness for prolonged use, surpassing approaches (e.g., AgentCF) that depend on recent-sampling strategies and require frequent profile updates. However, an exciting direction for future work involves exploring the optimal duration for which these precomputed personas retain their efficacy in online deployment. Understanding the dynamics of performance degradation over time can inform strategies for adaptive persona updates.

Ethics

We use publicly available datasets collected under standard ethical protocols and strictly adhere to their intended research use. PersonaX is designed solely for academic purposes, and by following these safeguards, we uphold ethical standards in data usage, privacy protection, and transparency.

Acknowledgments

This work was sponsored by the Australian Research Council under the Linkage Projects Grant LP210100129.

References

Wenruo Bai and Jeff Bilmes. 2018. Greed is still good: Maximizing monotone submodular+supermodular (bp) functions. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 304–313. PMLR.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 2318–2335, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. 2023. Lift yourself up: Retrieval-augmented text generation with self memory. *Preprint*, arXiv:2305.02437.

Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*, RecSys '23, page 1126–1132, New York, NY, USA. Association for Computing Machinery.

Chengcheng Guo, Bo Zhao, and Yanbing Bai. 2022. Deepcore: A comprehensive library for coreset selection in deep learning. In *International Conference on Database and Expert Systems Applications*, pages 181–195. Springer.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. In *European Conference on Information Retrieval*, pages 364–381. Springer.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE.

Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. 2023. Do llms understand user preferences? evaluating llms on user rating prediction. *Preprint*, arXiv:2305.06474.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: communicative agents for "mind" exploration of large language model society. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. *Preprint*, arXiv:2304.10149.

Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. 2024. Once: Boosting content-based recommendation with both open- and closed-source large language models. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, page 452–461, New York, NY, USA. Association for Computing Machinery.

Sichun Luo, Bowei He, Haohan Zhao, Yinya Huang, Aojun Zhou, Zongpeng Li, Yuanzhang Xiao, Mingjie Zhan, and Linqi Song. 2023. Recranker: Instruction tuning large language model as ranker for top-k recommendation. *arXiv preprint arXiv:2312.16018*.

Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Chris Leung, Jiajie Tang, and Jiebo Luo. 2024. LLM-rec: Personalized recommendation via prompting large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 583–612, Mexico City, Mexico. Association for Computational Linguistics.

Kai Mei and Yongfeng Zhang. 2023. Lightlm: a lightweight deep and narrow language model for generative recommendation. *arXiv preprint arXiv:2310.17488*.

Kai Mei, Xi Zhu, Wujiang Xu, Wenyue Hua, Mingyu Jin, Zelong Li, Shuyuan Xu, Ruosong Ye, Yingqiang Ge, and Yongfeng Zhang. 2024. Aios: Llm agent operating system. *arXiv:2403.16971*.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *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 188–197.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. 2021. Deep learning on a data diet: Finding important examples early in training. *Advances in Neural Information Processing Systems*, 34:20596–20607.

Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Practice on long sequential user behavior modeling for click-through rate prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2671–2679, New York, NY, USA. Association for Computing Machinery.

Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 2685–2692, New York, NY, USA. Association for Computing Machinery.

Ameya Prabhu, Philip H. S. Torr, and Puneet K. Dokania. 2020. Gdumb: A simple approach that questions our progress in continual learning. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II*, page 524–540, Berlin, Heidelberg. Springer-Verlag.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis. *Preprint*, arXiv:2307.16789.

Haocong Rao, Cyril Leung, and Chunyan Miao. 2023. Can ChatGPT assess human personalities? a general evaluation framework. *arXiv preprint arXiv:2303.01248*.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. 2017. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

Xubin Ren and Chao Huang. 2024. Easyrec: Simple yet effective language models for recommendation. *Preprint*, arXiv:2408.08821.

Chris Richardson, Yao Zhang, Kellen Gillespie, Sudipta Kar, Arshdeep Singh, Zeynab Raeesy, Omar Zia Khan, and Abhinav Sethy. 2023. Integrating summarization and retrieval for enhanced personalization via large language models. In *Proceedings of the First Workshop on Personalized Generative AI (PGAI '23), co-located with the 32nd ACM International Conference on Information and Knowledge Management (CIKM 2023)*, Birmingham, United Kingdom. Association for Computing Machinery. Workshop Paper.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2024. LaMP: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7370–7392, Bangkok, Thailand. Association for Computational Linguistics.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295.

Yunfan Shao, Linyang Li, Junqi Dai, and Xipeng Qiu. 2023. Character-LLM: A trainable agent for role-playing. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13153–13187, Singapore. Association for Computational Linguistics.

Yunxiao Shi, Min Xu, Haimin Zhang, Xing Zi, and Qiang Wu. 2024a. A learnable agent collaboration network framework for personalized multimodal ai search engine. *Preprint*, arXiv:2409.00636.

Yunxiao Shi, Xing Zi, Zijing Shi, Haimin Zhang, Qiang Wu, and Min Xu. 2024b. Enhancing retrieval and managing retrieval: A four-module synergy for improved quality and efficiency in rag systems. In *ECAI 2024*, pages 2258–2265. IOS Press.

Yunxiao Shi, Xing Zi, Zijing Shi, Haimin Zhang, Qiang Wu, and Min Xu. 2024c. Eragent: Enhancing retrieval-augmented language models with improved accuracy, efficiency, and personalization. *Preprint*, arXiv:2405.06683.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari S Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. *arXiv preprint arXiv:2206.14486*.

Guangzhi Sun, Xiao Zhan, and Jose Such. 2024. Building better ai agents: A provocation on the utilisation of persona in llm-based conversational agents. In *Proceedings of the 6th ACM Conference on Conversational User Interfaces*, CUI '24, New York, NY, USA. Association for Computing Machinery.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. 2019. An empirical study of example forgetting during deep neural network learning. *Preprint*, arXiv:1812.05159.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D. Nguyen. 2025. Multi-agent collaboration mechanisms: A survey of llms. *Preprint*, arXiv:2501.06322.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Lei Wang, Jingsen Zhang, Hao Yang, Zhiyuan Chen, Jiakai Tang, Zeyu Zhang, Xu Chen, Yankai Lin, Ruihua Song, Wayne Xin Zhao, Jun Xu, Zhicheng Dou, Jun Wang, and Ji-Rong Wen. 2024b. User behavior simulation with large language model based agents. *Preprint*, arXiv:2306.02552.

Zefan Wang, Zichuan Liu, Yingying Zhang, Aoxiao Zhong, Jihong Wang, Fengbin Yin, Lunting Fan, Lingfei Wu, and Qingsong Wen. 2024c. Rcagent: Cloud root cause analysis by autonomous agents with tool-augmented large language models. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 4966–4974, New York, NY, USA. Association for Computing Machinery.

Max Welling. 2009. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1121–1128.

Wujiang Xu, Shaoshuai Li, Mingming Ha, Xiaobo Guo, Qiongxu Ma, Xiaolei Liu, Linxun Chen, and Zhenfeng Zhu. 2023. Neural node matching for multi-target cross domain recommendation. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 2154–2166. IEEE.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025a. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*.

Wujiang Xu, Yunxiao Shi, Zujie Liang, Xuying Ning, Kai Mei, Kun Wang, Xi Zhu, Min Xu, and Yongfeng Zhang. 2025b. iagent: Llm agent as a shield between user and recommender systems. In *Findings of the Association for Computational Linguistics ACL 2025*.

Wujiang Xu, Qitian Wu, Zujie Liang, Jiaojiao Han, Xuying Ning, Yunxiao Shi, Wenfang Lin, and Yongfeng Zhang. 2024a. Slmrec: empowering small language models for sequential recommendation. In *The Thirteenth International Conference on Learning Representations*.

Wujiang Xu, Qitian Wu, Runzhong Wang, Mingming Ha, Qiongxu Ma, Linxun Chen, Bing Han, and Junchi Yan. 2024b. Rethinking cross-domain sequential recommendation under open-world assumptions. In *Proceedings of the ACM Web Conference 2024*, pages 3173–3184.

Rui Yang, Lin Song, Yanwei Li, Sijie Zhao, Yixiao Ge, Xiu Li, and Ying Shan. 2023. Gpt4tools: Teaching large language model to use tools via self-instruction. *Preprint*, arXiv:2305.18752.

An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. 2024a. On generative agents in recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 1807–1817, New York, NY, USA. Association for Computing Machinery.

Junjie Zhang, Yupeng Hou, Ruobing Xie, Wenqi Sun, Julian McAuley, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2024b. Agentcf: Collaborative learning with autonomous language agents for recommender systems. In *Proceedings of the ACM Web Conference 2024*, WWW '24, page 3679–3689, New York, NY, USA. Association for Computing Machinery.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024a. Expel: Llm agents are experiential learners. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pages 19632–19642. AAAI Press.

Qingfei Zhao, Ruobing Wang, Yukuo Cen, Daren Zha, Shicheng Tan, Yuxiao Dong, and Jie Tang. 2024b. Longrag: A dual-perspective retrieval-augmented generation paradigm for long-context question answering. *Preprint*, arXiv:2410.18050.

Qinlin Zhao, Jindong Wang, Yixuan Zhang, Yiqiao Jin, Kaijie Zhu, Hao Chen, and Xing Xie. Competeai: Understanding the competition dynamics of large language model-based agents. In *Forty-first International Conference on Machine Learning*.

Haizhong Zheng, Rui Liu, Fan Lai, and Atul Prakash. 2023. Coverage-centric coreset selection for high pruning rates. *Preprint*, arXiv:2210.15809.

Yujia Zhou, Qiannan Zhu, Jiajie Jin, and Zhicheng Dou. 2024. Cognitive personalized search integrating large language models with an efficient memory mechanism. In *Proceedings of the ACM Web Conference 2024*, WWW '24, page 1464–1473, New York, NY, USA. Association for Computing Machinery.

Xi Zhu, Yu Wang, Hang Gao, Wujiang Xu, Chen Wang, Zhiwei Liu, Kun Wang, Mingyu Jin, Linsey Pang, Qingsong Weng, Philip S. Yu, and Yongfeng Zhang. 2025. Recommender systems meet large language model agents: A survey. *Foundations and Trends® in Privacy and Security*, 7(4):247–396.

Contents

APPENDIX

A Time Complexity Analysis

In this section, we provide a rigorous analysis of the time complexity of user modeling in AgentCF and Agent4Rec, and examine how these complexities change when each method leverages PersonaX.

A.1 Preliminary Analysis

We begin by analyzing the time complexity of two sampling approaches. **Recent sampling**: This approach selects the most recent k user behaviors, which requires $O(1)$ computational complexity. Let $\mathcal{F}$ be the hardware's floating-point throughput, hence Recent sampling's time complexity in seconds is $O(1/\mathcal{F})$. **Relevance Strategy**: This strategy identifies user behaviors most pertinent to the target item I_{target}. Encoding an item into a feature vector using a large language embedding model incurs a time complexity of $O(d)$, and thus encoding n items results in a complexity of $O(nd)$. Selecting the top k most relevant items requires $O(n \log k/\mathcal{F})$, leading to an overall complexity of $O(nd + n \log k/\mathcal{F})$.

A.2 Analysis of LLM-UM methods

Summarization: In this method, the short behavioral sequence (SBS) $\mathcal{S}^*$ is distilled into a user representation $\mathcal{P}(\mathcal{S})$, and its complexity is independent of the sequence length k. Thus, the overall complexity remains $O(T)$. **Reflection**: This method iteratively updates the user persona along with $\mathcal{S}^*$. In the best case, all inferences are correct on the first attempt, incurring a complexity of $O(kT)$. In the worst case, all initial inferences fail, requiring a single reflection to update the persona, which enables the second inference to be correct. This results in a complexity of $O(3kT)$. Taking an average across these cases yields an approximate complexity of $O(2kT)$. Given that commonly $k \leq 10$, this constant factor remains manageable.

A.3 Analysis for AgentCF and Agent4Rec

AgentCF (Recent + Reflection). As this method updates the user profile dynamically with new behaviors, it is not well suited for offline profiling and be cached for long-term usage. The user profile is constructed once with a complexity of $O(1) + O(2kT)$. The profile is reused for inferring N_I items, each requiring $O(T)$. Thus the overall online complexity is $O(1/\mathcal{F} + 2kT + N_I T)$.

AgentCF (Relevance + Reflection). Each item in the user's behavior sequence is embedded with a complexity of $O(nd)$. The user profile is constructed with a complexity of $O(n \log k) + O(2kT)$. For each inference, the complexity is $O(d + T)$. Thus the overall online complexity is $N_I \cdot O(n \log k/\mathcal{F} + 2kT + d + T)$.

Agent4Rec (Relevance + Summarization). Each item is embedded with a complexity of $O(nd)$. The user profile is constructed once with a complexity of $O(n \log k) + O(T)$. Each inference has a complexity of $O(d + T)$. Thus the overall online complexity is $N_I \cdot O(n \log k/\mathcal{F} + d + 2T)$.

Agent4Rec (Recent + Summarization). As this method updates the user profile dynamically, it is not suited for offline profiling. The user profile is constructed once with a complexity of $O(1) + O(T)$. Thus the overall online complexity is $O(1/\mathcal{F} + T + N_I T)$.

Agent4Rec+PersonaX. Item embedding incurs $O(nd)$. The sampling process has a complexity of $O(\text{Cluster} + A.1 + A.2)$. Multiple persona generation requires $O(CT)$. The overall offline complexity is $O(CT + nd + \text{Cluster} + A.1 + A.2)$. For online phase, retrieving the user profile incurs $O(d) + O(1)$, and each inference requires $O(T)$. Thus the overall online complexity is $N_I \cdot O(T + 1/\mathcal{F} + d)$.

AgentCF+PersonaX. Item embedding incurs $O(nd)$. The sampling process has a complexity of $O(\text{Cluster} + A.1 + A.2)$. Multiple persona generation requires $O(C \cdot 2kT)$. The overall offline complexity is $O(C \cdot 2kT + nd + \text{Cluster} + A.1 + A.2)$. For online phase, retrieving the user profile incurs $O(d) + O(1)$, and each inference requires $O(T)$. Thus the overall online complexity is $N_I \cdot O(T + 1/\mathcal{F} + d)$.

Since the terms $O(1/\mathcal{F})$ and $O(n \log k/\mathcal{F})$ are asymptotically negligible compared to $O(T)$, they can be safely omitted. Thus we can get the results presented in Table 1.

Table 4: Summary of preprocessed subset statistics. "Avg.L" represents the average length of user behavior sequences.

Subsets	#Users	#Items	#Inters	Sparsity	Avg.L
$\texttt{CDs}_{50}$	100	4,899	5,000	98.97%	50.00
$\texttt{CDs}_{200}$	1000	101,902	200,336	99.80%	200.34
$\texttt{Books}_{480}$	1000	222,539	481,455	99.78%	481.46

B Datasets

In this appendix, we provide a detailed description of the dataset construction and statistics.

Building on prior studies such as AgentCF (Zhang et al., 2024b), Agent4Rec (Zhang et al., 2024a), and EasyRec (Ren and Huang, 2024), we evaluate our proposed method using two widely adopted subsets of the Amazon review dataset (Ni et al., 2019): *CDs and Vinyl* and *Books*. For the CDs dataset, we construct $\texttt{CDs}_{50}$, and $\texttt{CDs}_{200}$, with average user interaction sequence lengths of 50 and 200, respectively. These settings are similar as those used in AgentCF (Zhang et al., 2024b).

For the Books dataset, departing from the approach of Agent4Rec which limits each user's interactions to 20 items, we follow the guidelines of (Pi et al., 2019, 2020) to construct longer interaction sequences. Specifically, we create $\texttt{Books}_{480}$, with average sequence lengths of 480, respectively. Detailed statistics for these datasets are provided in Table 4.

Due to the high computational cost and expense associated with API calls for GPT-4o-mini, we conduct each experiment only once per dataset to ensure feasibility within a reasonable budget. This approach is common in agent recommendation studies (Zhang et al., 2024a,b; Wang et al., 2024b; Luo et al., 2023) and large-scale recommendation system research. Moreover, the larger number of users (1000) in our study enhances the reliability of the experimental results.

Note that we apply different LLM-UM methods to each dataset: Reflection for CDs50, and Summarization for CDs200 and Books480. The reason is that Reflection becomes inefficient as sequence length grows—a limitation also noted in the original AgentCF, and Summarization is more suitable for longer behavior sequence.

C Backbone Methods

We provide a detailed description of the backbone methods used for validation.

AgentCF (Zhang et al., 2024b) employs a reflective mechanism to model user personas. In the original framework, both the user profile and item profile are dynamically updated. In our implementation, the item profile is textually represented by concatenating the item's fields, while the user profile is initially set to "Currently Unknown" and is iteratively refined through continuous reflection. Furthermore, for the downstream recommendation ranking task in AgentCF, we replace the original LLM-based ranking with the EasyRec framework (Ren and Huang, 2024). EasyRec is the first large language embedding model specifically designed for recommendation. It aligns textual semantic spaces with collaborative behavioral signals, enabling recommendation tasks to rely solely on textual instructions (e.g., user preference descriptions and item profiles) while achieving performance comparable to traditional state-of-the-art models. Leveraging EasyRec for point-wise ranking is more experimentally efficient, accurate, and robust compared with LLMs.

Agent4Rec (Zhang et al., 2024a) maintains an agent profile comprising two key components: social traits and unique tastes. In our implementation, we streamline the process by focusing solely on capturing diverse user interests through the construction of unique tastes, thus simplifying experimentation. To achieve this, we adopt the summarization method from the original work, which distills user preferences from their behavioral sequences. Additionally, we replace the original rating prediction task in the Agent4Rec framework with a ranking task.

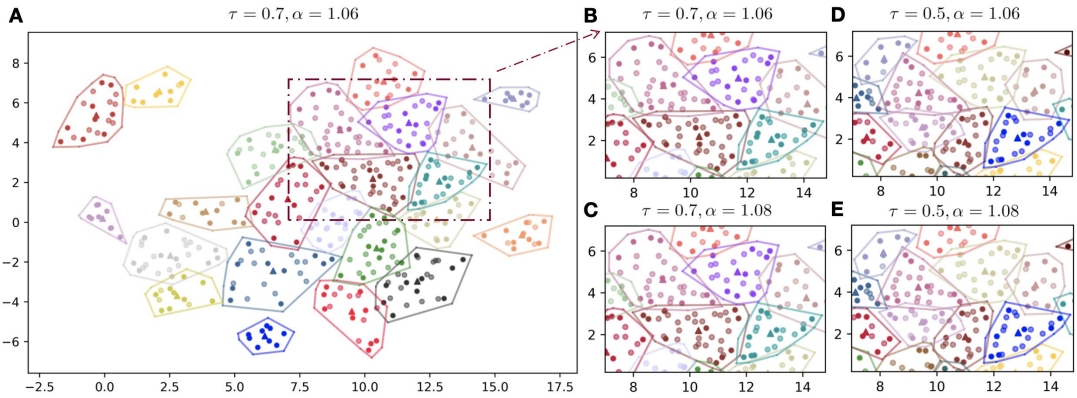

Figure 5: Sampling process for a user in $\texttt{Books}_{480}$ with a 50% selection ratio. Points are color-coded and outlined. Non-transparent points signify data selected, whereas transparent points delineate behaviors not sampled. A offers a holistic perspective on the user's comprehensive behavior distribution, capturing the full extent of engagement patterns. BẼ presents parts of behaviors distributions and sampling process under varying configurations of hyper-parameters. Triangles denote the centroids of the clusters.

D Hyper-parameter Analysis and Sampling Process Visualization

This section delves into the influence of the hyperparameters τ and α on the performance of PersonaX, as they play pivotal roles in shaping the hierarchical clustering and in-cluster behavior selection processes. Specifically, τ dictates the granularity of the hierarchical clustering. A larger τ value yields coarser clusters, encompassing a broader spectrum of behavioral samples with potentially greater divergence from the cluster centroid. In contrast, a smaller τ enforces a more stringent clustering criterion, resulting in finer-grained clusters characterized by higher intra-cluster homogeneity. On the other hand, α modulates the balance between prototypicality and diversity during the in-cluster behavior selection stage. A higher α amplifies the preference for selecting behavior samples further from the cluster centroid, thereby enhancing diversity within the cluster. Conversely, a lower α emphasizes prototypicality, favoring samples that closely align with the cluster centroid. Our empirical analysis, as illustrated in Figure 4, uncovers nuanced patterns in how these hyperparameters influence the model's overall performance.

1. Performance at Low Ratios: Across τ and α configurations, the performances at lower ratios (e.g., 0.1, 0.3) remain similar. This is because the selected samples at low ratios primarily originate near the cluster centroid, regardless of the diversity adjustment imposed by α. Slightly superior performance of $\tau = 0.5$ compared to $\tau = 0.7$ at these ratios is attributed to the finer clustering granularity of $\tau = 0.5$, which ensures that selected samples exhibit higher prototypicality.

2. Performance at High Ratios (0.5–0.9): At higher ratios, configurations with larger α values (e.g., $\alpha = 1.06, 1.08$) outperform their smaller-α counterparts (e.g., $\alpha = 1.01, 1.04$). This highlights the efficacy of the in-cluster selection strategy: after a core set of prototypical samples is chosen, incorporating more diverse samples significantly enhances performance. The inclusion of diversity helps capture broader behavioral patterns, leading to improved generalization.

3. Trade-offs in Specific Settings: A nuanced behavior is observed in the interaction between τ and α. For $\tau = 0.5$, $\alpha = 1.08$ performs better than $\alpha = 1.06$, suggesting that in scenarios where the cluster scope is relatively constrained, the diversity of samples becomes pivotal, necessitating a higher α to effectively prioritize and capture heterogeneous behaviors. For $\tau = 0.7$, $\alpha = 1.06$ outperforms $\alpha = 1.08$, as the broader cluster scope with $\alpha = 1.08$ potentially overemphasizes highly diverse samples, leading to a slight degradation in overall performance. This interplay underscores the importance of balancing cluster granularity and diversity during sample selection.

4. Parameter Robustness: Our framework demonstrates robust performance across a wide range of hyper-parameter settings. For instance, the worst best performance (71.6) achieved with $\tau = 0.7, \alpha = 1.04$ is only marginally lower than the best performance of the relevance baseline (71.86). This indicates that our method remains effective without being overly sensitive to hyper-parameter adjustments.

To provide an intuitive analysis of the sampling process, we conducted a visualization study, as illustrated in Figure 5. From Figure 5.A, it is evident that smaller clusters are preferentially allocated an adequate sampling quota compared to larger ones. This observation underscores the efficacy of the proposed Algorithm 1, which strategically prioritizes smaller clusters to ensure sufficient sampling. By adopting this approach, the algorithm effectively preserves the user's diverse interests, including long-tail preferences, even under constrained sampling resources. The comparisons between Figure 5.B and Figure 5.C, as well as Figure 5.D and Figure 5.E, highlight the impact of α. Specifically, smaller α values tend to focus the sample selection closer to the cluster centroids. Furthermore, the comparisons between Figure 5.B and Figure 5.D, and between Figure 5.C and Figure 5.E, demonstrate that a a more granular clustering can constrain Algorithm 2 from selecting samples that deviate excessively from the cluster centroids. This constraint mitigates potential performance degradation caused by overemphasis on unrelated samples.

The experimental findings and visualization analysis suggest that both τ and α require empirical tuning to identify optimal configurations. We recommended a balance between prototypicality and diversity, for example a larger α values combined with appropriately tuned small τ.

E Details about In-Cluter Selection

In this section, we delve into the mechanisms governing sample selection by proposing a principled scoring system to evaluate the prototypicality and diversity of candidate samples. The scoring mechanism is derived from two complementary perspectives: prototypicality

$$\frac{1}{1 + d(\mathbf{e}_j, \mu_i)}$$

, which assesses how representative a sample is of its respective cluster, and diversity

$$\frac{2}{a_i} \sum_{\substack{I_a, I_b \in c_i^* \\ a \neq b}} d(\mathbf{e}_a, \mathbf{e}_b)$$

, which quantifies the extent to which the selected samples span a broader spectrum of the data distribution.

E.1 Prototypicality and Diversity Scoring

From the formulation below,

$$\max_{c_i^*} \left(w_p \cdot \sum_{I_j \in c_i^*} \frac{1}{1 + d(\mathbf{e}_j, \mu_i)} + w_d \cdot \frac{2}{a_i} \sum_{\substack{I_a, I_b \in c_i^* \\ a \neq b}} d(\mathbf{e}_a, \mathbf{e}_b) \right)$$

it is evident that the prototypicality score exhibits an inverse relationship with the distance between a sample and the center of its cluster. As a sample moves further from the cluster centroid, its prototypicality diminishes proportionally, reflecting its reduced ability to represent the typical characteristics of the cluster. The diversity score considers the pairwise distances between the candidate sample and the samples already selected. This ensures that the inclusion of a new sample enriches the diversity of the chosen subset by discouraging redundancy.

To compute the diversity score, we employ the scaling factor $2/a_i$. We don't choice of averaging scaling approach $1/[a_i(a_i - 1)]$, which tends to normalize diversity growth. By adopting $2/a_i$, we deliberately amplify the influence of diversity as a_i increases, thereby prioritizing the inclusion of diverse samples in scenarios where a cluster is allocated enough sampling budget. This design reflects an underlying intent: as a_i grows, the system places greater emphasis on diversity to ensure comprehensive coverage of the data distribution. Conversely, when a_i is small, prototypicality takes precedence, directing attention toward selecting samples that are most representative of their respective clusters.

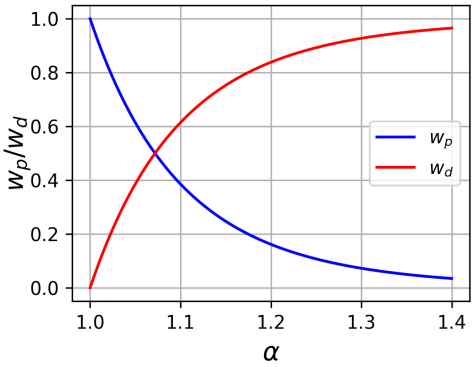

Figure 6: Trade off between w_p and w_d at different settings of α.

E.2 Design Rationale

The decision to amplify diversity dynamically aligns with our broader goal of achieving a balanced and adaptive sample selection process. By coupling prototypicality with diversity in this manner, we address two critical challenges in data selection:

1. **Representative Sampling**: When the sample pool is sparse, selecting highly prototypical samples ensures that the chosen subset faithfully captures the core characteristics of the data clusters. This is particularly crucial in tasks where the representativeness of the selected data has a direct impact on model performance, such as user profiling or content recommendation.

2. **Comprehensive Coverage**: In cases where the candidate pool is dense, diversity becomes increasingly important to avoid redundancy and to capture the subtle variations within the data distribution. By amplifying diversity when a_i is large, our scoring mechanism ensures that the selected subset spans the breadth of the distribution, enabling downstream models to generalize better across diverse scenarios.

E.3 Broader Implications

The proposed scoring framework introduces a novel perspective on balancing representativeness and diversity in data selection. By dynamically modulating the influence of diversity based on the local sample density, our approach strikes a principled balance between selecting typical and atypical samples. This adaptability is particularly valuable in data-centric applications, where sample selection directly affects the quality of downstream tasks, such as dataset pruning, user interest modeling, and few-shot learning.

E.4 Visualization Explanation

Figure 6 shows the trade-off between w_p and w_d across different settings of α. As observed in the figure, when α is small, w_p dominates the sampling process, leading to the selection of samples near the cluster center. These samples are prototypical and reflect the representative thematic interests of the cluster. As α increases, w_d becomes more prominent, and w_p approaches 0, causing the sampling process to prioritize diverse samples in order to enhance generalization.

Figure 7 presents a dynamic visualization of the sampling process in Algorithm 2. As illustrated, the algorithm iteratively selects samples by jointly optimizing for both prototypicality and diversity, thereby maximizing the combined gain. This approach stands in contrast to conventional data selection methods, which often exhibit a unimodal bias—either favoring simple, centrally clustered, and highly representative samples (Welling, 2009; Rebuffi et al., 2017; Sorscher et al., 2022) or prioritizing difficult, outlier samples with strong generalization potential (Paul et al., 2021; Toneva et al., 2019).

Empirical analysis of the hyperparameter α, which governs the trade-off between prototypicality and diversity, reveals a practical range of 1.06–1.08. Within this regime, PersonaX often firstly selects a minimal set of prototypical samples and then shifting its focus toward maximizing sample diversity. We believe this is because of the superior few-shot generalization capabilities of LLMs. These models inherently require fewer prototypical instances to capture core user interests, thereby shifting their emphasis toward diverse sample acquisition to further enhance generalization.

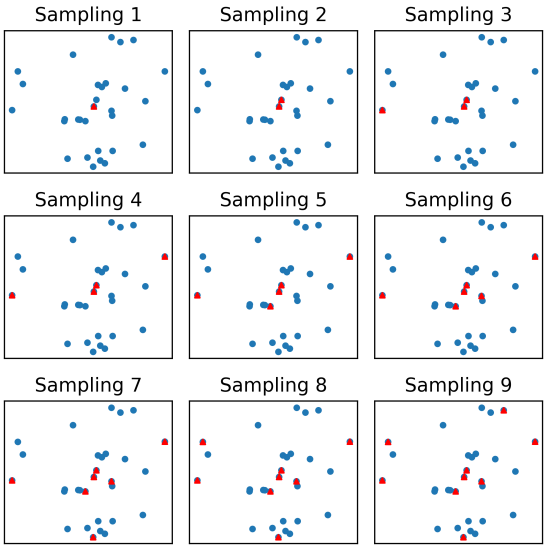

Figure 7: Dynamics of In-Cluster sample selection: We set $\tau = 0.7$, with the samples distributed within the range of $[-2, 2]$.

Table 5: Intuitive comparison of user personas modeled by different methods. Specifically, (A) Relevance sampling (length 3), (B) Recent sampling (length 10), and (C) PersonaX (selection ratio 30%).

Method	**User Persona**	**o3-mini's Reasoning Evaluation**
A	I prefer Rock and Progressive Rock music, particularly works by artists like Trey Anastasio. I enjoy albums that feature new songs and live performances, as they provide a dynamic listening experience. I appreciate behind-the-scenes content that adds depth to the music experience. I also enjoy Contemporary Folk music, especially works by artists like Steve Winwood, as they can offer a fresh take on musical compositions. I dislike R&B and Soul music, particularly collections that focus on remixes of older tracks, as I find them less engaging. I prefer original compositions and fresh takes on music rather than revisiting past favorites. Additionally, I am not particularly interested in World Music, especially live performances that do not align with my preferred genres.	This description is detailed and rich in information, but it introduces multiple music genres which may dilute the focus on the user's core preferences.
B	I prefer Classic Rock and Arena Rock music, particularly albums that showcase strong melodies and engaging performances, such as 'Chickenfoot' by Chickenfoot. I enjoy music that has a fun, energetic vibe and often features collaborative efforts from talented musicians. I appreciate emotional ballads and songs that reflect depth and complexity in lyrics and composition. I dislike Blues Rock that focuses on traditional guitar work and may lack the innovative sounds I seek, as exemplified by 'Smokestacks, Broom Dusters & Hoochie Coochie Men' by Micky Moody, which I find less appealing due to its more conventional approach.	This description focuses on a subset of rock music—Classic and Arena Rock—which contrasts with the broader rock and progressive preferences seen in the other descriptions. It is detailed but less aligned with the core focus compared to C.
C	I prefer rock and progressive music, particularly works by notable artists like Trey Anastasio. I enjoy albums that offer a collection of new songs, especially those that include additional content such as live performances and behind-the-scenes footage. I dislike pop and dance music, particularly generic albums that lack depth or a compelling narrative. I appreciate immersive listening experiences that connect me to the artist's journey and creative process.	This description is the most concise and focused, effectively capturing the user's core interests—new material, live performances, and behind-the-scenes insights—without extraneous details, making it the highest quality among the three.

F Theoretical Analysis for In-cluster Selection

Algorithm 2 (*P*rototypicality–*D*iversity balanced *S*ub-*B*ehaviour *S*equence) selection greedily builds an SBS $S_t \subseteq V_i$ of cardinality t by adding at each iteration the element with the largest marginal gain with respect to the mixed objective (1). We establish three properties: monotonicity, finite termination, and a

Table 6: Quantitative Evaluation of User Persona Modeling Methods

Method	NDCG@1	NDCG@5	NDCG@10	Hit@1	Hit@5	Hit@10	MRR
A	0.00	0.42	0.54	0.00	0.67	1.00	0.39
B	0.00	0.54	0.54	0.00	1.00	1.00	0.39
C	0.33	0.71	0.71	0.33	1.00	1.00	0.61

data–dependent 94.79% approximation guarantee.

F.1 Objective Function Properties

The following is the in-cluster SBS selection objective function for a fixed cluster c_i^*:

$$\max_{c_i^*} f(c_i^*) = w_p \underbrace{\sum_{I_j \in c_i^*} \frac{1}{1 + d(\mathbf{e}_j, \boldsymbol{\mu}_i)}}_{f_p(c_i^*)} + w_d \underbrace{\frac{2}{a_i} \sum_{\substack{I_a, I_b \in c_i^* \\ a \neq b}} d(\mathbf{e}_a, \mathbf{e}_b)}_{f_d(c_i^*)}, \tag{1}$$

where $d(\cdot, \cdot)$ denotes a metric in the embedding space, $\boldsymbol{\mu}_i$ is the centroid of cluster i, a_i is sampling size, and $w_p, w_d \geq 0$ are fixed weights.

Definition 1. Let V be a finite ground set. A set function $f\colon 2^V \to \mathbb{R}$ is *submodular* if for all $A \subseteq B \subseteq V$ and for every $e \in V \setminus B$,

$$f(A \cup \{e\}) - f(A) \;\geq\; f(B \cup \{e\}) - f(B). \tag{2}$$

A set function is *supermodular* when the inequality is reversed, and *modular* when equality always holds.

Lemma 1. *The prototypicality component f_p in (1) is modular and hence submodular.*

Proof. For any subset $S \subseteq V$,

$$f_p(S) = \sum_{I_j \in S} \frac{1}{1 + d(\mathbf{e}_j, \boldsymbol{\mu}_i)} = \sum_{I_j \in S} g(I_j),$$

where $g(I_j)$ depends solely on the singleton I_j. Thus the marginal gain of adding e is always $g(e)$, independent of S, satisfying the equality condition in Definition 1. □

Lemma 2. *The diversity component f_d in (1) is supermodular.*

Proof. Let $A \subseteq B \subseteq V$ and $e \in V \setminus B$. Denote $m_A(e) = f_d(A \cup \{e\}) - f_d(A)$ and $m_B(e) = f_d(B \cup \{e\}) - f_d(B)$. By direct expansion,

$$m_A(e) = \frac{2}{a_i} \sum_{a \in A} d(\mathbf{e}, \mathbf{e}_a)\,, \qquad m_B(e) = \frac{2}{a_i} \sum_{b \in B} d(\mathbf{e}, \mathbf{e}_b)\,.$$

Because $A \subseteq B$, every term in the sum for $m_A(e)$ appears in $m_B(e)$ (and $m_B(e)$ contains additional non-negative terms due to metric non-negativity). Hence $m_A(e) \leq m_B(e)$, which is exactly the reverse inequality of Definition 1, establishing supermodularity. □

Proposition 1. *The full objective $f = w_p f_p + w_d f_d$ is the weighted sum of a modular (submodular) function and a supermodular function. Consequently f itself is* neither *submodular nor supermodular unless $w_d = 0$ or $w_p = 0$, respectively.*

Proof. Immediate from Lemmas 1 and 2 and the linearity of set functions. □

Monotonicity. Every candidate element has non-negative marginal gain, the greedy rule—selecting at iteration t the element with the largest marginal improvement—yields a sequence of objective values $f(S_{t+1}) > f(S_t)$ until the prescribed cardinality a_i is reached.

Table 7: Pointwise ratios for the supermodular (g) and submodular (f) components; lower ratios imply higher curvature. Empirically SBS selection only has $a_i \leq 5$, we **highlight** the values with attaining the minimum.

Point	$r_v^g = \frac{g(v)}{g(v \mid V \setminus \{v\})}$	$r_v = \frac{f(v \mid V \setminus \{v\})}{f(v)}$
1	0.0765	0.1648
2	0.0822	0.2075
3	0.0666	0.1509
4	**0.0493**	**0.1159**
5	0.0884	0.2137
6	0.0599	0.1374
7	0.0847	0.2028
8	0.0301	0.0915
9	0.0851	0.2056
10	0.0194	0.0696

Finite Termination. The procedure performs exactly a_i iterations, inserting one element per step; therefore it terminates after a finite number of steps.

F.2 Performance of the Greedy Algorithm

Bian et al. (Bai and Bilmes, 2018) study the maximisation of monotone *BP* functions—sums of a monotone submodular component f and a monotone supermodular component g—under a cardinality constraint. Let[2]

$$\kappa_g \;=\; 1 - \min_{v \in V} \frac{g(v)}{g\big(v \mid V \setminus \{v\}\big)}, \qquad\qquad \kappa_f \;=\; 1 - \min_{v \in V} \frac{f\big(v \mid V \setminus \{v\}\big)}{f(v)}. \tag{3}$$

The quantities $\kappa_g \in [0,1]$ and $\kappa_f \in [0,1]$ are termed the *curvatures* of g and f, respectively, and capture how far each component deviates from modularity. For monotone BP functions the simple greedy algorithm that, at every step, adds the element of highest marginal gain enjoys the worst-case guarantee.

$$\frac{F_{\text{greedy}}}{F^\star} \;\geq\; \frac{1}{\kappa_f}\Big(1 - \exp\big(-\kappa_f(1-\kappa_g)\big)\Big), \tag{4}$$

where $F^\star$ is the optimal objective value.

Empirical curvatures. Using the settings $\tau = 0.7$ and $\alpha = 1.06$, we compute the pointwise ratios $r_v^g = \frac{g(v)}{g(v|V\setminus\{v\})}$ and $r_v = \frac{f(v|V\setminus\{v\})}{f(v)}$ for the ten most representative items. Table 7 reports the results (smaller ratios—*bold* in the table—produce larger curvatures).

Taking the minima over all points gives $\kappa_g \leq 0.9806$ and $\kappa_f \leq 0.9304$. Substituting these into (4) yields a *guaranteed* approximation factor of 94.79% for our in-cluster greedy selector (Algorithm 2).

Conclusion. Despite the lack of submodularity, the curvature-aware guarantee shows that the proposed greedy in-cluster selection is near-optimal in practice.

G Case Study

In this section, we present a case study comparing user personas modeled using Relevance, Recent, and PersonaX methods, with the backbone LLM-UM approach fixed as Reflection. The dataset used is $\texttt{CDs}_{50}$, with the User ID `A2NQUGGYM0DBM1`. The results are summarized in Table 5. We evaluate their quality by OpenAI's o3-mini, using its reasoning capabilities in an LLM-As-Judge framework. The evaluation indicated that Model C had the highest modeling quality [3]. The explanation provided was that

[2]We denote $f(v) = f(\{v\})$ and $f(v \mid S) = f(S \cup \{v\}) - f(S)$ for brevity.

[3]Repeated inquiries occasionally resulted in A being rated higher, with the justification that A offered a more comprehensive view. However, this comprehensiveness came at the cost of interest modeling that was more diffuse and less precise.

C demonstrated superior descriptive quality, capturing the user's core preferences for rock and progressive music with concise and precise language. It also emphasized the user's interest in new releases, live performances, and behind-the-scenes content, while avoiding extraneous information misaligned with primary interests. In contrast, Model A, while rich in information, introduced a broader range of music styles that diluted focus, and Model B predominantly emphasized an alternative style of rock, leading to inconsistencies with the other descriptions.

We conducted three rounds of quantitative evaluations on the ranking task, each comprising one positive item alongside nigh negative items. As shown in Table 6, Method C achieved the highest performance, followed by Method B, while Method A exhibited the poorest performance.

H Prompt Templates

We present the prompt templates used in AgentCF, as shown in Figure 8 and Figure 9, and those employed in Agent4Rec, depicted in Figure 10.

Prompt Template for Forward Inference Process of AgentCF

Task: We provide a user's personal profile in [User Profile], which includes the user's preferences, dislikes, and other relevant information. You need play the role of the user. And we also provide two candidate items, A and B, with their features in [Item Feature]. You need to choice between the two item candidates based on your profile and the features of the items. Furthermore, you must articulate why you've chosen that particular item while rejecting the other.
User Profile: {profile}
Item Feature: Item A: {item a} Item B: {item b}
Steps to Follow:
1. Extract your preferences and dislikes from your self-introduction.
2. Evaluate the two candidate in light of your preferences and dislikes. Make your choice by considering the correlation between your preferences/dislikes and the features of the candidates.
3. Explain why you made such choices, from the perspective of the relationship between your preferences/dislikes and the features of these candidate items.
Important Notes:
1. Your output should strictly be in the following format: Chosen Item: Item A or Item B Explanation: Your detailed rationale behind your choice and reasons for rejecting the other item.
2. When identifying user's likes and dislikes, do not fabricate them! If your [User Profile] doesn't specify any relevant preferences or dislikes, use common knowledge to inform your decision.
3. You **must** choose one of these two candidates, and **cannot** choose both.
4. Your explanation needs to be comprehensive and specific. Your reasoning should delve into the finer attributes of the items.
5. Base your explanation on facts. For instance, if your self-introduction doesn't reveal any specific preferences or dislikes, justify your decision using available or common knowledge.
6. Please ignore the effect of Item position and length, they do not affect your decision.
Response Example: *Chosen Item: Item A Explanation: I chose Item A because...*

Figure 8: Prompt template for the forward process of AgentCF to predict one user potentially liked item between a positive one and a negarive one.

Prompt Template for Backward Reflection Process of AgentCF

Background: We provide a user's personal profile in [User Profile], which includes the user's preferences, dislikes, and other relevant information. You need play the role of the user. Recently, you considered choosing one more prefered Item from two candidates. The features of these two candidate are provided in [Item Feature]. And your choice and explanation is in [Choice and Explanation], which reveals your previous judgment for these two candidates.
User Profile: {profile}
Item Feature: Item A: {item a} Item B: {item b}
Choice and Explanation: {response}
Task: However, The user in the real world actually prefer to choose Item B, and reject the Item A that you initially chose. This indicates that you made an incorrect choice, the [Choice and Explanation] was mistaken. Therefore, you need to reflect and update [User Profile].
Steps to Follow:
1. Analyze the misconceptions in your previous [Choice and Explanation] about your preferences and dislikes, as recorded in your explanation, and correct these mistakes.
2. Explore your new preferences based on the Item B you really enjoy, and determine your dislikes based on the Item a you truly don't enjoy.
3. Summarize your past preferences and dislikes from your previous [User Profile]. Combine your newfound preferences and dislikes with your past ones. Filter and remove any conflicting or repetitive parts in your past [User Profile] that contradict your current preferences and dislikes.
4. Generate a update profile use the following format:
My updated profile: {Please write your updated profile here}
Important Notes:
1. Keep your updated profile under 180 words.
2. Any overall assessments or summarization in your profile are forbidden.
3. Your updated profile should only describe the features of items you prefer or dislike, without mentioning your wrong choice or your thinking process in updating your profile.
4. Your profile should be specific and personalized. Any preferences and dislikes that cannot distinguish you from others are not worth recording.
Response Example: *My updated profile: I ...*

Figure 9: Prompt template for the backward process of AgentCF to apply the reflect mechanism for updating user profile.

Prompt Template for Summarization Process of Agent4Rec

Task: We provide a user's personal profile in [User Profile], which includes the user's preferences and other relevant information. Additionally, we provide a sequence of liked items in [Sequence Item Profile] that the user has interacted with. Your task is to analyze these items in the context of the user's existing profile and produce an updated profile that reflects any new preferences, or insights inferred from the user's interactions with these items.
User Profile: {profile}
Sequence Item Profile: {sequence item profile}
Steps to Follow:
1. Carefully review the user's existing profile to understand their stated preferences and dislikes.
2. Analyze the features of the items in the provided sequence, noting any common themes, attributes, or patterns.
3. Identify any new preferences that can be inferred from the user's interactions with these items.
4. Summarize and update the user's profile by incorporating the new insights, adding new preferences or dislikes, and highlighting any changes or developments in the user's tastes. Important Notes
5. Your output should strictly be in the following format: Summarization: {Your updated profile.}
6. Do not contradict the user's existing preferences unless there is clear evidence from the sequence items that their tastes have changed.
7. Base your summary on facts and logical inferences drawn from the items in the sequence.
8. Be comprehensive and specific in your summarization, focusing on the finer attributes and features of the items that relate to the user's preferences.
9. Avoid fabricating any information not supported by the user's profile or the sequence items.
Response Example: Summarization: You've developed interest in ....

Figure 10: Prompt template of Agent4Rec to apply the summarization mechanism for distilling user profile.