ULMRec: User-centric Large Language Model for Sequential Recommendation

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have demonstrated promising performance in sequential recommendation, leveraging their superior language understanding capabilities. However, most existing LLM-based recommendation models primarily capture sequential patterns between items and overlook the nuanced nature of individual user preferences, i.e., users with similar interaction histories demonstrate different interests. To alleviate this limitation, in this paper, we propose ULMRec, a framework that effectively integrates user personalized preferences into LLMs for sequential recommendation. For integrating the user personalized preference, we design two key components: (1) user indexing: a personalized user indexing mechanism that leverages vector quantization on user reviews and user IDs to generate meaningful and unique user representations, and (2) alignment tuning: an alignment-based tuning stage that employs comprehensive preference alignment tasks to enhance the model's capability for capturing personalized information. In this way, ULMRec achieves deeper integration of language semantics with user personalized preferences, facilitating effective adaptation to recommendation. Extensive experiments on two public datasets demonstrate that ULMRec significantly outperforms existing methods, validating the effectiveness of our approach. The code is available at `https://anonymous.4open.science/r/ULMRec`.

1 Introduction

As the dynamic nature of user interests and behavioral patterns, sequential recommendation (Kang and McAuley, 2018; Sun et al., 2019) has attracted significant attention recently. Most existing sequential recommendation methods adopt various neural networks to capture item co-occurrence patterns. Recently, Large Language Models (LLMs) (Touvron et al., 2023) have opened new frontiers in recommender systems (Yue et al., 2023; Zheng et al., 2024) by leveraging their advanced semantic understanding and pre-trained knowledge. For instance, LlamaRec (Yue et al., 2023) leverages LLMs to re-rank candidate items, demonstrating their effectiveness in capturing complex user-item semantic relationships. These LLM-based methods have shown superior performance. However, most methodologies primarily focus on the modeling of item-to-item relationships (i.e., sequential patterns of which items frequently co-occur together in user interactions), while failing to adequately capture user-specific preference patterns. In other words, these methods may not effectively differentiate between users who, despite having similar interaction histories, exhibit distinct preferences, which is illustrated in Figure 1.

To bridge this gap, we propose developing an effective framework that incorporates user personalized preference into LLMs. A straightforward approach would be to integrate user IDs, which serve as unique user identifiers, into LLMs by prefixing them to users' historical behaviors during the fine-tuning process. However, this naive integration faces two significant challenges:

(1) Semantic gap. A significant disconnect exists between the language semantics modeled by LLMs and the preference information embedded in user IDs within recommender systems. This disconnect occurs because LLMs are trained on natural language and may not recognize the special significance of user IDs in a recommendation context. When LLMs tokenize user IDs, they inadvertently fragment and potentially destroy the inherent personalized preference information encoded within the identifiers.

(2) Limited task. Fine-tuning LLMs solely on the next-item prediction task may constrain the model to learn merely superficial item co-occurrence patterns in users' historical sequences, rather than developing a comprehensive understanding of users' personalized preferences. This narrow focus would

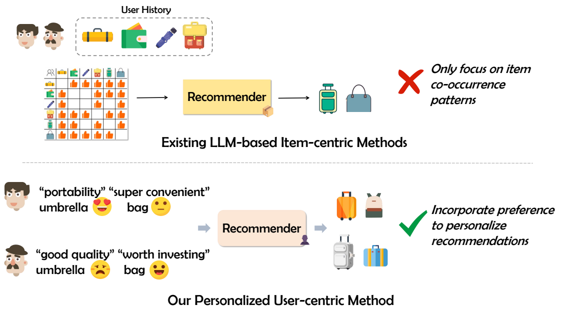

User History
Recommender
Only focus on item co-occurrence patterns
Existing LLM-based Item-centric Methods
"portability" "super convenient"
umbrella
bag
"good quality" "worth investing"
umbrella
bag
Recommender
Incorporate preference to personalize recommendations
Our Personalized User-centric Method

Figure 1: Illustration of two methods for LLM-based recommenders: item-centric and user-centric.

not capture the fine-grained user personalized preferences, which extend beyond simple sequential patterns. Hence, we tackle this integration problem in two main aspects:

(1) User Indexing: We develop an innovative allocation mechanism that creates meaningful user indices,designed to both preserve user preferences and maintain unique representations.

(2) Alignment Tuning: We design a sophisticated alignment strategy that integrates language semantics with user preferences, extending beyond basic next-item prediction to capture comprehensive user behavior patterns.

In this paper, we propose ULMRec, an LLM-based model that enhances recommendation from a user-centric perspective by integrating language semantics with personalized user preferences. Our model operates in two main phases: personalized user indexing stage and alignment-based index understanding stage. In the first stage, we leverage vector quantization (Lee et al., 2022) to generate personalized user indices based on historical reviews, which contain rich information about user interests. Unlike traditional linear or MLP mappings that focus on individual user-item interactions, vector quantization effectively captures user collaborative preference patterns by discretizing user behaviors into shared interest clusters. To ensure uniqueness and semantic meaningfulness, we incorporate original user IDs into the quantization process, effectively preventing index collisions. Although our generated indices successfully encode personalized user information and avoid conflicts with the LLM's existing token, a key challenge remains: how to enable the LLM to accurately interpret and utilize these indices. To address this, we expand beyond traditional sequential recommendation tasks and design a comprehensive set of preference alignment tasks in the second stage, incorporating diverse personalized signals including user preferences, historical behaviors, and rating patterns. These alignment tasks work collaboratively to enhance the model's capability in understanding the personalized user interests. In all, the contributions of our work can be summarized as:

- We propose ULMRec, which could bridge the semantic gap between LLMs and personalized recommendation by integrating user preferences into LLMs. It is the first attempt to address the fundamental challenge of incorporating user-level personalization into LLMs.
- We develop a personalized user indexing mechanism and combine it with carefully designed alignment tasks. Our approach enables LLMs to deeply understand fine-grained user preferences, providing more comprehensive and detailed user modeling.
- Extensive experiments on two public datasets demonstrate that our approach significantly outperforms existing methods, validating the effectiveness of our user-centric personalization injection strategy.

2 Related Work

2.1 Large Language Model

Large Language Models (LLMs) have revolutionized natural language processing in recent years. The development of LLMs can be traced through several key milestones. Initially, models like BERT (Devlin et al., 2019) introduced the concept of pre-training on vast amounts of text data, enabling better understanding of language context. A significant leap came with GPT-3 (Brown, 2020; Ouyang et al., 2022), which demonstrated remarkable few-shot learning capabilities across various tasks. Despite their remarkable capabilities, LLMs face significant challenges when applied to personalized recommender systems. The reason is that these models excel at general language understanding and generation tasks, but they struggle to capture fine-grained user preferences crucial for effective recommendations.

2.2 Sequential Recommendation

Sequential recommendation predicts users' interests by modeling sequential patterns in user-item interactions. Various neural architectures like RNN (Li et al., 2017), MLP (Zhou et al., 2022), CNN (Tang and Wang, 2018; Yuan et al., 2019), GNN (Wu et al., 2019; Chang et al., 2021) and Transformer (Kang and McAuley, 2018; Sun et al., 2019) have been employed to capture item co-occurrence patterns in interaction sequences.

With the emergence of Large Language Models (LLMs) like Llama (Touvron et al., 2023), researchers have begun integrating them into recommender systems (Bao et al., 2023; Cui et al., 2022; Yang et al., 2023; Geng et al., 2022). Common approaches convert user behaviors into text sequences and design prompts for recommendation tasks. For instance, TallRec (Bao et al., 2023) structures recommendation data as instructions, while LC-Rec (Zheng et al., 2024) uses tree-structured vector quantization for item indexing.

However, existing LLM-based methods primarily focus on modeling sequential patterns and item co-occurrence (e.g., which products are frequently viewed or purchased together), often failing to adequately address the complex and nuanced nature of individual user preferences that extend beyond simple interaction sequences. Our work aims to bridge this gap by integrating LLMs with traditional recommender systems from a user modeling perspective, thereby not only leveraging item co-occurrence patterns but also deeply understanding and incorporating personalized user preferences.

3 Problem Statement

Sequential recommendation aims to predict users' future interests based on their historical interactions. Formally, we define the problem as follows: Let $\mathcal{U} = [u_1, u_2, ..., u_{|\mathcal{M}|}]$ denotes the set of users and $\mathcal{I} = [i_1, i_2, ..., i_{|\mathcal{N}|}]$ denotes the set of items, where $\mathcal{M}$ and $\mathcal{N}$ represent the number of users and items respectively. For each user $u \in \mathcal{U}$, their interaction history is represented as a sequence $S_u = [i_1, i_2, ..., i_t]$, where $i_k \in \mathcal{I}$ and t is the sequence length. Each interaction is associated with a timestamp, and the sequence is ordered chronologically. Given a user u and their interaction history S_u, the goal of sequential recommendation is to predict the next item i_{t+1} that the user is most likely to interact with.

4 Proposed Method

In this section, we will introduce ULMRec in detail. To bridge the gap between language semantics and user personalized semantics effectively, we propose a two-stage framework that addresses this challenge from a user-centric perspective, as illustrated in Figure 2:

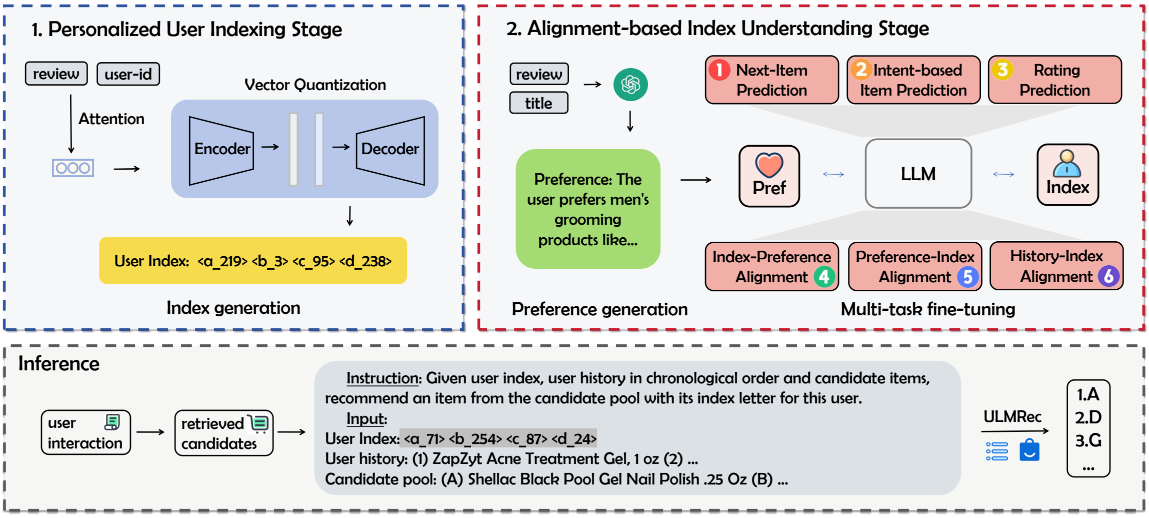
Personalized User Indexing Stage. In this stage, we encode users' preferences into unique hierarchical semantic IDs through vector quantization of user reviews and personalized information. The learned indices preserve both collaborative relationships among similar users and distinguishable features for individuals.

Alignment-based Index Understanding Stage. To establish semantic connections between user indices and concrete preference information, we design a comprehensive set of preference alignment tasks beyond traditional next-item prediction. Through instruction tuning, these tasks guide LLMs to understand the semantic meaning behind each index and align them with specific user preferences.

4.1 Personalized User Indexing

The key to introducing the user personalized preference into the LLMs is how to represent the user with index IDs and integrate these IDs into LLMs. Firstly, directly using original user IDs poses a risk of semantic conflicts with LLM's token semantics. Secondly, although constructing user profiles through LLM-generated descriptions offers an alternative approach, it often fails to capture the nuanced and individualistic aspects of user preferences. In this section, we propose an approach that leverages vector quantization techniques (Zeghidour et al., 2021) to generate user indices. This method enables the encapsulation of personalized preference information in a format compatible with LLM architectures.

To generate semantic user indices with rich preference information, we leverage user-generated reviews as they provide valuable insights into individual preferences. For each user $u \in \mathcal{U}$, we first collect their historical reviews as a sequence $W_u = [w_1, w_2, ..., w_n]$, where n represents the number of reviews. We then employ BERT (Devlin et al., 2019) to transform these reviews into high-dimensional embeddings: $\mathbf{E}_u = BERT(W_u) = [\mathbf{e}_{w_1}, \mathbf{e}_{w_2}, ..., \mathbf{e}_{w_n}]$. To enhance user representation and avoid potential collisions, we also obtain the embedding $\mathbf{e}_{o_u}$ of user's original ID o_u in the same way as reviews and integrate it through an attention mechanism. The attention process can be formu-


1. Personalized User Indexing Stage
review
user-id
Attention
Vector Quantization
Encoder
Decoder
User Index: <a_219> <b_3> <c_95> <d_238>
Index generation
2. Alignment-based Index Understanding Stage
review
title
Preference: The user prefers men's grooming products like...
Preference generation
1 Next-Item Prediction
2 Intent-based Item Prediction
3 Rating Prediction
Pref
LLM
Index
Index-Preference Alignment 4
Preference-Index Alignment 5
History-Index Alignment 6
Multi-task fine-tuning
Inference
user interaction
retrieved candidates
ULMRec
1.A
2.D
3.G
...

Instruction: Given user index, user history in chronological order and candidate items, recommend an item from the candidate pool with its index letter for this user.
Input:
User Index: <a_71> <b_254> <c_87> <d_24>
User history: (1) ZapZyt Acne Treatment Gel, 1 oz (2) ...
Candidate pool: (A) Shellac Black Pool Gel Nail Polish .25 Oz (B) ...

Figure 2: The framework of ULMRec. During training, we first construct personalized user indices via vector quantization, then fine-tune the LLM through multi-task learning to understand these indices that encode user-specific representations. During inference, we leverage the tuned LLM to evaluate the recommendation performance.

lated as:

$$\alpha_i = softmax(\mathbf{e}_{w_i}^T \mathbf{A}_i \mathbf{e}_{o_u}), \mathbf{x}_u = \sum_{i=1}^{n} \alpha_i \mathbf{e}_{w_i}, \tag{1}$$

where $\mathbf{A}_i$ is a learnable weight matrix, α_i is the attention weight for the i-th review, and $\mathbf{x}_u$ is the aggregated review representation.

Then, using it as input, we train a Residual-Quantized Variational AutoEncoder (RQ-VAE) (Zeghidour et al., 2021) to hash the user information into discrete personalized semantic IDs in a unified space. Through its hierarchical latent spaces, this model can effectively capture complex non-linear relationships. Specifically, RQ-VAE encodes the input user embedding $\mathbf{x}$ to obtain a latent representation $\mathbf{z}$. At the initial level (l = 0), residual is defined as $\mathbf{z}$. And in each level l, we have a codebook $C^l = \{\mathbf{v}_k^l\}_{k=1}^K$, where $\mathbf{v}_k^l$ represents the codebook vector and K is the codebook size. At the 0-th level, we map the latent representation $\mathbf{r}_0 = \mathbf{z}$ to the closest vector $\mathbf{v}_k^0$ in codebook C^0, where the index of it is the 0-th codeword d_0. Then, at the next level, the residual vector is computed as: $\mathbf{r}_1 = \mathbf{r}_0 - \mathbf{v}_{d_0}^0$. This process iteratively finds the closest embedding in each codebook level to get p codewords as the hierarchical semantic IDs. Finally, the sum of each quantized codebook vector is an approximation of the original input vector. Below describes this process, where d_i represents the index of the closest embedding, namely the i-th codeword of user indices.

$$d_i = \arg\min_k \left\| \mathbf{r}_i - \mathbf{v}_k^i \right\|_2^2, \tag{2}$$

$$\mathbf{r}_{i+1} = \mathbf{r}_i - \mathbf{v}_{d_i}^i. \tag{3}$$

After generating the semantic IDs, the quantized representation of $\mathbf{z}$, computed as: $\hat{\mathbf{z}} = \sum_{i=0}^{p-1} \mathbf{v}_{d_i}^i$, is used as the decoder input to re-construct the input user embedding $\mathbf{x}$. The training loss contains reconstruction loss and RQ loss, which are defined as follows:

$$\mathcal{L}_{\text{RECON}} = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2, \tag{4}$$

$$\mathcal{L}_{\text{RQ}} = \sum_{i=0}^{p-1} \left\| \text{sg}\left[\mathbf{r}_i\right] - \mathbf{v}_{d_i}^i \right\|_2^2 + \beta \left\| \mathbf{r}_i - \text{sg}\left[\mathbf{v}_{d_i}^i\right] \right\|_2^2, \tag{5}$$

$$\mathcal{L}_{\text{RQ-VAE}} = \mathcal{L}_{\text{RECON}} + \mathcal{L}_{\text{RQ}}. \tag{6}$$

4.2 Alignment-based Index Understanding

Although user semantic IDs are constructed, the LLMs often struggle to fully grasp their intrinsic meanings. To better integrate the user semantic IDs into the LLMs, we design a series of customized tasks aimed at aligning these indices with personalized user semantics. This approach encourages LLMs to comprehensively learn about the underlying personal preferences from provided context. To mitigate catastrophic forgetting in LLM training, we employ a cross-task data shuffling strategy that promotes balanced learning by maintaining exposure to diverse training examples.

4.2.1 Next-Item Prediction

The first instruction tuning task focuses on next-item prediction. We construct prompts by combining three key elements: the user index, a chronological sequence of historical interactions, and a set of previously retrieved candidate items. LLMs are enforced to predict the next item that a user is most likely to interact with from the candidates. This task enables the LLM to develop its core recommendation capabilities. However, to sufficiently integrate personalized user preferences, additional deep semantic alignment tasks are necessary.

4.2.2 Index-Preference Alignment

To explicitly align user indices with their preference information, we employ two processes: (1) Preference extraction: We utilize GPT-3.5-Turbo (Brown, 2020) to extract user preferences from reviews, which reflect users' attitudes and tastes comprehensively; (2) Index-preference alignment: To enhance LLMs' understanding and inference capabilities regarding user indices, we instruct them to reconstruct user preferences based solely on the user index. In this way, the LLM could effectively interpret and utilize these indices in subsequent tasks, grounding them in actual user preferences and behaviors.

4.2.3 Preference-Index Alignment

In order to further enhance the LLMs' ability to understand user preferences, we reverse the input and output of the index-preference alignment task. This reversed task serves as a counterpart to the earlier index-to-preference mapping, creating a bidirectional understanding of the relationship between user indices and preferences. With this support, the model could better associate user preferences with their respective indices in a bidirectional learning process.

4.2.4 History-Index Alignment

Users with similar interaction histories may have distinct preferences, for example, one user might prioritize price while another focuses on quality. To address this nuance, we design a task that aligns historical behaviors with unique user indices. This alignment aims to couple each user's interaction history with their distinctive index, enabling LLMs to differentiate between users with similar behaviors but divergent preferences.

4.2.5 Rating Prediction

To explicitly capture users' preferences towards specific items better, we incorporate the rating score for deep alignment. In this task, we provide the LLM with user index, preference, history, and past ratings. Notably, the rating for the last interacted item is omitted, leaving it for the LLM to predict. This task enhances the LLM's ability to understand and predict fine-grained user preferences, contributing to more personalized recommendations.

4.2.6 Intent-based Item Prediction

User preferences generated via GPT contain rich intent information, potentially benefiting prediction. We hypothesize that the LLM could interpret user intentions from these preferences and make informed recommendations. Then we design a task in which the LLM is provided with user preferences derived from "preference extraction" and candidate items, instructing them to decode user interests and select the most probable preferred item. This approach aims to capture nuanced user intentions beyond observable behaviors.

4.3 Model Training and Inference

During training, we first utilize LRURec (Yue et al., 2024) retriever to obtain Top-20 candidate items for each user-item interaction. Then, we employ Llama-2-7b (Touvron et al., 2023) as the backbone model. For the task of predicting the next item, we follow LlamaRec (Yue et al., 2023) to use index letters to identify candidates and employ the verbalizer to transform LLM outputs into ranking scores over these candidates.

As the objectives of all alignment tasks are to generate tokens based on the given context, we directly employ the cross-entropy loss of the generation target as follows:

$$\mathcal{L} = -\sum_{i=1}^{m} y_i \log\left(\hat{y}_i\right), \qquad (7)$$

where m denotes the vocabulary size, y_i represents the actual token, and $\hat{y}_i$ represents the predicted probability. In our experiments, we adopt QLoRA (Dettmers et al., 2024) to perform quantization on model parameters for efficient training.

During inference, we evaluate the ULMRec performance on the next-item prediction task with the user index and the user historical behaviors. Besides, we extracts logits using the verbalizer to rank relevant items in the candidate item pool.

5 Experiment

5.1 Experimental Settings

We conduct our experiments on two widely used Amazon datasets that are popular for LLM training in recommendation: **Beauty** and **Video Games** (He and McAuley, 2016; McAuley et al., 2015). Following previous work (Yue et al., 2023; Zheng et al., 2024), we filter out users with fewer than five interactions. The detailed statistics of these processed datasets are presented in Table 1.

Dataset	# Users	# Items	# Interact.	# Length	# Sparsity
Beauty	22,332	12,086	198k	8.87	99.9%
Games	15,264	7,676	148k	9.69	99.8%

Table 1: Statistics of the datasets.

We evaluate our model performance against two categories of baseline models: (1) Classical sequential recommendation models: **NARM** (Li et al., 2017), **BERT4Rec** (Sun et al., 2019), **SASRec** (Kang and McAuley, 2018), **LRURec** (Yue et al., 2024), **Llama-2** (Touvron et al., 2023), and **LlamaRec** (Yue et al., 2023). (2) Recent representative LLM-based sequential recommendation models: a. Prompt-based methods: **P5** (Geng et al., 2022), **POD** (Li et al., 2023), and **PeaPOD** (Ramos et al., 2024); b. Item-indexing based methods: **TIGER** (Rajput et al., 2024), **CID+IID** (Hua et al., 2023), **TransRec** (Lin et al., 2024), and **IDGenRec** (Tan et al., 2024); c. User-profile based methods: **PALR** (Yang et al., 2023), **RDRec** (Wang et al., 2024), and **P2Rec** (Liu et al., 2024).

In evaluation, we adopt three widely used metrics: Mean Reciprocal Rank (**MRR@k**), Normalized Discounted Cumulative Gain (**NDCG@k**) and Recall (**Recall@k**). In our experiment, we set k as 5 and 10. Besides, in the ranking stage, we perform evaluation on the retrieved subset and then combine the ranking metrics with retrieval performance as the overall metrics.

5.2 Performance Comparison

5.2.1 Main Performance

As shown in Table 2, we evaluate the performance of ULMRec and other baseline methods both on valid retrieval subsets and entire datasets. From the results, we can observe: (1) Among traditional sequential recommendation models, SASRec consistently outperforms NARM and BERT4Rec across all metrics, underscoring the effectiveness of self-attention mechanisms in capturing sequential dependencies; (2) LLM-based methods show promising results in recommendation tasks over traditional methods, particularly in the Games dataset. LlamaRec achieves the best results among baselines, indicating the potential of LLMs in recommendation scenarios; (3) Our proposed ULMRec demonstrates consistent and substantial improvements over all baselines. Taking the Games dataset as an example, ULMRec achieves relative improvements of 8.7%, 8.4%, and 7.7% in R@5, R@10, and M@10 respectively, compared to the strongest baseline LlamaRec, validating the effectiveness of our user-centric learning paradigm design.

5.2.2 Further Comparison with LLM-based Methods

Furthermore, we compare our model with recent representative LLM-based methods in Table 3. Similar to LlamaRec, we compare against the reported results from their original papers. From the results, we can find: (1) P5, POD and PeaPOD attempt to incorporate user/item IDs directly in LLMs. Due to the interpretation difficulty of discrete IDs in LLMs and the potential loss of collaborative preference information during tokenization their improvements remain limited; (2) TIGER, CID+IID, TransRec and IDGenRec focus on enhancing sequential recommendation through item indexing methods. However, by primarily focusing on item-related information, these methods neglect the high-order user personalized preference, which is also crucial to recommendation; (3) PALR, RDRec and P2Rec have explored different approaches to incorporate user information through prompts, preference generation, and category distribution modeling. However, these methods lack a unique identifier for users, which may cause confusion in distinguishing different user personalized preferences; (4) In contrast, our ULMRec addresses these limitations through hierarchical user indexing and comprehensive preference alignment. Experimental results show significant improvements over the second best baseline, achieving 32%, 19%, 16%, and 8% improvements in R@10, R@5, N@10, and N@5 respectively.

5.3 Ablation Study

To further explore how different alignment tasks impact the model performance, we conduct ablation experiments with six variants (Games dataset as examples), including: (1) w/o pref: removing the index-preference alignment task; (2) w/o intention:

Method		Ranking						Overall					
		M@5	N@5	R@5	M@10	N@10	R@10	M@5	N@5	R@5	M@10	N@10	R@10
Games	NARM	0.2039	0.2424	0.3600	0.2248	0.2931	0.5168	0.0479	0.0576	0.0874	0.0541	0.0729	0.1351
	BERT4Rec	0.1765	0.2109	0.3160	0.1947	0.2551	0.4530	0.0422	0.0512	0.0788	0.0478	0.0649	0.1214
	SASRec	0.2177	0.2571	0.3776	0.2408	0.3134	0.5521	0.0515	0.0617	0.0930	0.0583	0.0783	0.1446
	Llama-2	0.2264	0.2720	0.4117	0.2558	0.3439	0.6352	0.0477	0.0574	0.0868	0.0539	0.0725	0.1339
	LRURec	0.2504	0.3009	0.4544	0.2811	0.3760	0.6879	0.0533	0.0640	0.0966	0.0598	0.0800	0.1463
	LlamaRec	0.2825	0.3360	0.4995	0.3158	0.4173	0.7522	0.0600	0.0714	0.1061	0.0671	0.0887	0.1599
	Our	**0.3071**	**0.3641**	**0.5379**	**0.3364**	**0.4354**	**0.7592**	**0.0647**	**0.0768**	**0.1134**	**0.0709**	**0.0918**	**0.1600**
Beauty	NARM	0.1961	0.2284	0.3263	0.2128	0.2689	0.4517	0.0289	0.0342	0.0503	0.0321	0.0420	0.0746
	BERT4Rec	0.1587	0.1901	0.2861	0.1743	0.2281	0.4043	0.0246	0.0298	0.0457	0.0276	0.0372	0.0686
	SASRec	0.2296	0.2679	0.3843	0.2491	0.3152	0.5312	0.0336	0.0397	0.0582	0.0371	0.0481	0.0844
	Llama-2	0.2617	0.3047	0.4360	0.2876	0.3687	0.6365	0.0344	0.0401	0.0574	0.0378	0.0485	0.0837
	LRURec	0.2944	0.3403	0.4801	0.3259	0.4168	0.7170	0.0376	0.0435	0.0614	0.0417	0.0533	0.0916
	LlamaRec	0.3016	0.3524	0.5071	0.3350	0.4337	**0.7600**	0.0385	0.0450	0.0648	0.0428	0.0554	0.0971
	Our	**0.3253**	**0.3792**	**0.5436**	**0.3520**	**0.4445**	0.7468	**0.0428**	**0.0499**	**0.0715**	**0.0463**	**0.0585**	**0.0982**

Table 2: Performance of Ranking and Overall. Ranking evaluates recommendations within the Top-20 retrieved items subset. Overall evaluates recommendations across the entire item space. The best and second-best results are in bold and underlined.

	P5	PALR	TIGER	CID+IID	TransRec	POD	PeaPOD	RDRec	P2Rec	IDGenRec	Our
N@5	0.0367	N/A	0.0321	0.0356	0.0365	0.0395	0.0445	0.0461	0.0445	0.0486	**0.0499**
R@5	0.0493	N/A	0.0454	0.0512	0.0504	0.0537	0.0588	0.0601	0.0604	0.0618	**0.0715**
N@10	0.0416	0.0446	0.0384	0.0427	0.0450	0.0443	0.0493	0.0504	0.0509	0.0541	**0.0585**
R@10	0.0645	0.0721	0.0648	0.0732	0.0735	0.0688	0.0738	0.0743	0.0852	0.0814	**0.0982**

Table 3: The overall performance compared to other LLM-based models on Beauty dataset.

removing the intent-based item prediction task; (3) w/o rating: removing the rating prediction task; (4) w/o history: removing the history-index alignment task; (5) w/o turnpref: removing the preference-index alignment task; (6) w/o all-align: removing all alignment tasks except next-item prediction.

Our ablation results are shown in Figure 3. From the results, we can find each alignment task contributes positively to recommendation performance, demonstrating the effectiveness of our designed comprehensive preference alignment. As expected, w/o all-align obtains the poorest performance, indicating that LLMs could not effectively capture user personalized preferences solely based on the next-item prediction task. Notably, w/o turnpref shows a worse performance compared to w/o pref. This may be because that w/o turnpref taking the user's personalized preferences as input makes it easier for LLMs to understand and generate the corresponding user index.

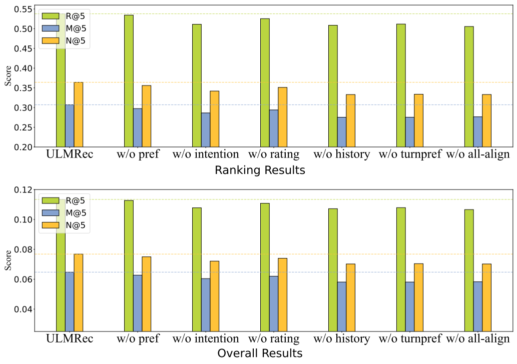


Figure 3: Ablation study of different alignment tasks.

5.4 Further Analysis

5.4.1 Index-only Recommendation

To evaluate our indexing method's capability in capturing user intentions, we conduct a specialized experiment called ULMRec-uid, where recommendations are made solely based on user indices without any chronological interaction history. This setup isolates and tests the model's ability to learn user semantic preferences purely through our indexing mechanism. We use Games dataset in the ranking phase as the example. As shown in Table 4, our model with aligned indices significantly outperforms traditional approaches like BERT4Rec, particularly in the R@10 metric (0.6647 vs. 0.4530), despite the absence of historical interaction data. These results provide strong evidence that ULMRec can effectively construct comprehensive user profiles and accurately capture personalized interests throught user indices alone, highlighting the potential of LLM-based approaches in scenarios where historical interaction data is limited or unavailable.

Model \ Metric	M@5	N@5	R@5	M@10	N@10	R@10
ULMRec	0.3071	0.3641	0.5379	0.3364	0.4354	0.7592
ULMRec-uid	0.1843	0.2330	0.3822	0.2209	0.3233	0.6647
BERT4Rec	0.1765	0.2109	0.3160	0.1947	0.2551	0.4530

Table 4: Performance comparison of index-only recommendation, where ULMRec-uid indicates generating recommendations solely based on the learned indices.

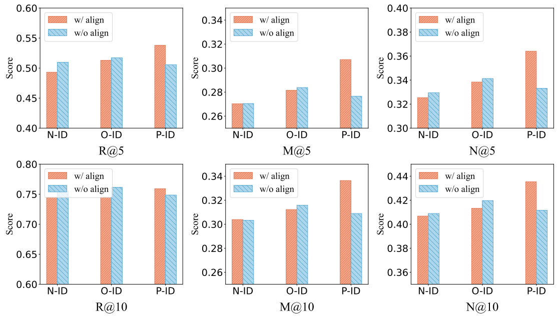


Figure 4: Performance comparison of different user indexing methods (N-ID: Numerical IDs, O-ID: Original IDs, P-ID: Personalized IDs) with and without alignment.

5.4.2 Analysis of Different User Indexing Methods

To systematically evaluate our proposed user indexing method, we compare three distinct approaches: (1) Numerical IDs (N-ID): Traditional numerical identifiers (e.g., 1, 2, 3) commonly used in recommender systems; (2) Original IDs (O-ID): Raw alphanumeric identifiers from the dataset (e.g., A1GNYV0RA0EQSS); (3) Personalized IDs (P-ID): Our proposed vector quantization approach that encodes user preferences into structured indices (e.g., <a_219> <b_2> <c_95> <d_238>). We further examine each method's performance with and without alignment tasks to understand their interaction with the LLM. As shown in Figure 4, our VQ-based method with alignment consistently achieves superior performance across all metrics, validating the effectiveness of our approach. Notably, both N-ID and O-ID perform better without alignment tasks. This could be attributed to LLMs interpreting these IDs as raw text, maintaining their original semantic understanding. The alignment process, in these cases, actually disrupts this inherent interpretation, leading to performance degradation. In contrast, our P-ID method, combined with carefully designed alignment tasks, successfully enables the LLM to integrate user preferences into the representations.

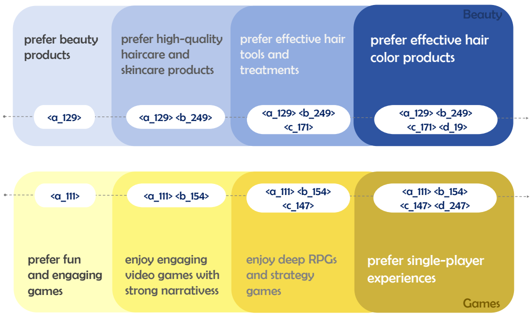

Beauty
prefer beauty products
prefer high-quality haircare and skincare products
prefer effective hair tools and treatments
prefer effective hair color products
<a_129>
<a_129> <b_249>
<a_129> <b_249> <c_171>
<a_129> <b_249> <c_171> <d_19>
<a_111>
<a_111> <b_154>
<a_111> <b_154> <c_147>
<a_111> <b_154> <c_147> <d_247>
prefer fun and engaging games
enjoy engaging video games with strong narrativess
enjoy deep RPGs and strategy games
prefer single-player experiences
Games

Figure 5: Case study for hierarchical preference evolution with multi-level user indices.

5.4.3 Complexity Analysis

In this section, we analyze the complexity of ULMRec. We employ GPT-3.5-Turbo to generate user preferences once per user, while fine-tuning and inference each require a single Llama-2 call per data point. Notably, ULMRec's efficiency is comparable to other LLM-based recommenders, averaging 0.615s per test instance on Beauty (vs. LlamaRec's 0.636s) and 0.423s on Games (vs. LlamaRec's 0.417s).

5.4.4 Case Study

To further explore the relationship between user preference and indices, we present two illustrative cases in Figure 5. We demonstrate how our model captures hierarchical user preferences through different levels of indices on both Beauty and Games datasets. As the level of indices increases, we observe that the preferences become progressively more specific, evolving from general beauty products to hair colors in Beauty dataset, and from general gaming interests to specific single-player experiences in Games dataset, enabling more precise and targeted recommendations.

6 Conclusion

In this paper, we introduce ULMRec, an LLM-based recommender that integrates user-item interactions and user personalized information into the LLMs. Our approach generates unique semantic user indices through vector quantization, then employs alignment tasks to incorporate user-specific preference semantics, which include sequential recommendation, explicit and implicit alignments, enabling LLMs to map indices to user characteristics and bridge semantic gaps across domains. Experiments demonstrate ULMRec's effectiveness in both indexing and alignment, outperforming state-of-the-art models in recommendation.

7 Limitation

While ULMRec exhibits competitiveness performance, we still observe some limitations of ULMRec. (1) Transferrable ability: The model would currently be limited to single-domain recommendations without cross-domain transfer capabilities. (2) Generation-recommendation trade-off: While the model achieves enhanced recommendation performance, it might compromise its inherent language generation capabilities.

References

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1007–1014.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2021. Sequential recommendation with graph neural networks. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 378–387.

Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. M6-rec: Generative pretrained language models are open-ended recommender systems. *arXiv preprint arXiv:2205.08084*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2019*, pages 4171–4186. Association for Computational Linguistics.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 299–315.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. 2023. How to index item ids for recommendation foundation models. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 195–204.

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

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. 2022. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532.

Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1419–1428.

Lei Li, Yongfeng Zhang, and Li Chen. 2023. Prompt distillation for efficient llm-based recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 1348–1357.

Xinyu Lin, Wenjie Wang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2024. Bridging items and language: A transition paradigm for large language model-based recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1816–1826.

Dugang Liu, Shenxian Xian, Xiaolin Lin, Xiaolian Zhang, Hong Zhu, Yuan Fang, Zhen Chen, and Zhong Ming. 2024. A practice-friendly llm-enhanced paradigm with preference parsing for sequential recommendation. *arXiv preprint arXiv:2406.00333*.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, and 1 others. 2024. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36.

Jerome Ramos, Bin Wu, and Aldo Lipani. 2024. Preference distillation for personalized generative recommendation. *arXiv preprint arXiv:2407.05033*.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450.

Juntao Tan, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Zelong Li, and Yongfeng Zhang. 2024. Idgenrec: Llm-recsys alignment with textual id learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 355–364.

Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 565–573.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Xinfeng Wang, Jin Cui, Yoshimi Suzuki, and Fumiyo Fukumoto. 2024. Rdrec: Rationale distillation for llm-based recommendation. *arXiv preprint arXiv:2405.10587*.

Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 346–353.

Fan Yang, Zheng Chen, Ziyan Jiang, Eunah Cho, Xiaojiang Huang, and Yanbin Lu. 2023. Palr: Personalization aware llms for recommendation. *arXiv preprint arXiv:2305.07622*.

Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2019. A simple convolutional generative network for next item recommendation. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 582–590.

Zhenrui Yue, Sara Rabhi, Gabriel de Souza Pereira Moreira, Dong Wang, and Even Oldridge. 2023. Llamarec: Two-stage recommendation using large language models for ranking. *arXiv preprint arXiv:2311.02089*.

Zhenrui Yue, Yueqi Wang, Zhankui He, Huimin Zeng, Julian McAuley, and Dong Wang. 2024. Linear recurrent units for sequential recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 930–938.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. 2021. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. 2024. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 1435–1448. IEEE.

Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Filter-enhanced mlp is all you need for sequential recommendation. In *Proceedings of the ACM web conference 2022*, pages 2388–2399.