ReLand: Integrating Large Language Models' Insights into Industrial Recommenders via a Controllable Reasoning Pool

Changxin Tian
tianchangxin.tcx@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Binbin Hu
bin.hbb@antfin.com
Ant Group
Hangzhou, Zhejiang, China

Chunjing Gan
cuibing.gcj@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Haoyu Chen
jieyun.chy@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Zhuo Zhang
zz297429@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Li Yu
jinli.yl@alibaba-inc.com
Ant Group
Hangzhou, Zhejiang, China

Ziqi Liu
ziqiliu@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Zhiqiang Zhang
lingyao.zzq@antfin.com
Ant Group
Hangzhou, Zhejiang, China

Jun Zhou*
jun.zhoujun@antfin.com
Ant Group
Hangzhou, Zhejiang, China

Jiawei Chen
sleepyhunt@zju.edu.cn
Zhejiang University
Hangzhou, Zhejiang, China

ABSTRACT

Recently, Large Language Models (LLMs) have shown significant potential in addressing the isolation issues faced by recommender systems. However, despite performance comparable to traditional recommenders, the current methods are cost-prohibitive for industrial applications. Consequently, existing LLM-based methods still need to catch up regarding effectiveness and efficiency. To tackle the above challenges, we present an LLM-enhanced recommendation framework named **ReLand**, which leverages **Re**trieval to effortlessly integrate **La**rge language models' insights into i**nd**ustrial recommenders. Specifically, ReLand employs LLMs to perform generative recommendations on sampled users (*a.k.a.,* seed users), thereby constructing an LLM Reasoning Pool. Subsequently, we leverage retrieval to attach reliable recommendation rationales for the entire user base, ultimately effectively improving recommendation performance. Extensive offline and online experiments validate the effectiveness of ReLand. Since January 2024, ReLand has been deployed in the recommender system of Alipay, achieving statistically significant improvements of 3.19% in CTR and 1.08% in CVR.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

*Corresponding author.

RecSys '24, October 14–18, 2024, Bari, Italy

ACM ISBN 979-8-4007-0505-2/24/10
https://doi.org/10.1145/3640457.3688131

KEYWORDS

Recommender Systems, Large Language Models

ACM Reference Format:
Changxin Tian, Binbin Hu, Chunjing Gan, Haoyu Chen, Zhuo Zhang, Li Yu, Ziqi Liu, Zhiqiang Zhang, Jun Zhou, and Jiawei Chen. 2024. ReLand: Integrating Large Language Models' Insights into Industrial Recommenders via a Controllable Reasoning Pool. In *18th ACM Conference on Recommender Systems (RecSys '24), October 14–18, 2024, Bari, Italy.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3640457.3688131

1 INTRODUCTION

In today's industrial applications, recommender systems have become increasingly crucial for enhancing user experience and driving business growth [30, 37]. However, traditional recommender systems are typically trained and deployed within isolated environments [44, 49], so they lack open-world knowledge and human-like insights. Notably, Large Language Models (LLMs) show remarkable progress in context understanding, reasoning and generalization, among other areas [52]. Recent studies [1, 29, 44, 48] suggest that LLMs possess significant potential for addressing the isolation issues faced by recommender systems in practical settings.

To enhance recommender systems with LLMs, pioneering studies have moved beyond simply using LLMs as feature encoders [22, 44]. They consider the recommendation task as instruction following [12, 29, 39, 48], prompting the LLMs to generate recommendation *rationales*, where the rationales specifically refer to reasoning texts that describe user preferences [14, 39]. However, most empirical findings [29, 49] indicate that relying on the zero-shot or few-shot learning capabilities of LLMs remains suboptimal when compared with traditional recommender systems, due to the absence of in-domain recommendation knowledge. To address this

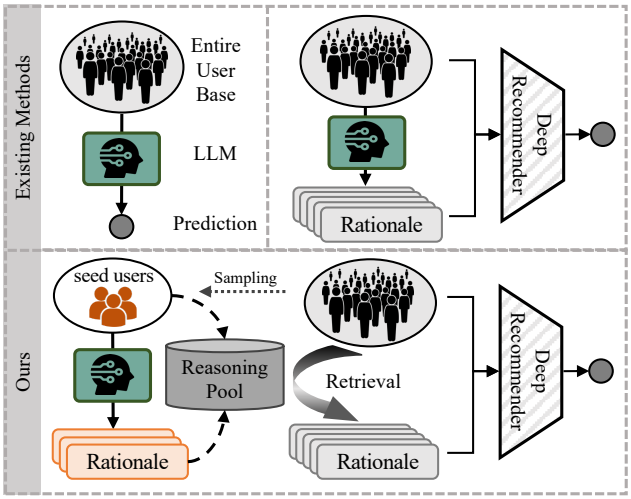


Figure 1: Comparison of existing LLM-based recommendation methods, including LLMs as recommenders (upper left) [1, 48] and LLMs as knowledge encoders (upper right) [44], and our framework ReLand (lower).

issue, increasing efforts [1, 23, 39, 49] propose to fine-tune LLMs using interaction data and in-domain information relevant to recommendation, successfully improving recommendation performance.

Despite significant advancements, when enhancing industrial-scale recommender systems with LLMs, there are challenges in terms of both effectiveness and efficiency. Regarding **effectiveness**, existing LLM-based recommenders rely on textual semantic understanding and commonsense-based reasoning to perform recommendations, which shows excellent performance in cold-start situations [1, 8, 12, 29]. However, in more typical scenarios, industrial recommender systems are capable of learning informative representations for users and items based on interaction data, which results in LLM-based methods underperforming well-trained traditional recommenders [20, 49]. Regarding **efficiency**, the immense scale of LLMs demands substantial computational resources, whereas industrial-scale recommender systems serve hundreds of millions of users and require millisecond-level response time. For instance, deploying the LLaMA-7B [34] model on a single NVIDIA A800 GPU and processing one hundred million users for recommendation is estimated to require approximately 1.518×10^4 hours[1]. Clearly, the cost of performing individual inferences for each user with any mainstream LLM is prohibitive. Although recent studies [16] leverage knowledge distillation to reduce LLMs' cost, they still face difficulties in processing industrial-scale user data daily. Thus, it remains a significant unresolved issue to efficiently enhance industrial-scale recommender systems with LLMs.

To tackle the above challenges, we present an LLM-enhanced recommendation framework named **ReLand**, that leverages **Re**trieval to effortlessly integrate **La**rge language models' insights into **ind**ustrial recommenders. Building on the assumption that users with similar behaviors can share the same recommendation rationale, we use LLMs as recommenders to perform generative recommendations on seed users, then utilize LLM-generated rationales to enhance the recommendation for the entire user base efficiently, as illustrated in Figure 1. Specifically, our framework encompasses two phases: extracting LLMs' insights with seed users and utilizing LLMs' insights via retrieval. (1) To handle the efficiency issue, in the first phase, we design an adaptive sampling strategy to select a small subset of users who mirror the diverse preferences and behavioral patterns of the whole user community, which we refer to as "*Seed Users*". Subsequently, we guide LLMs to step-by-step analyze the historical interactions of seed users, thereby extracting high-quality, reusable recommendation rationales from the LLM to construct an "*LLM Reasoning Pool*." (2) To handle the effectiveness issue, in the second stage, we match each user with a seed user who exhibits the most similar behavioral patterns. Then, we "*retrieve*" rationales associated with the matched seed users from the LLM Reasoning Pool, and repurposes these for the entire user base. Consequently, all users can attach reliable LLM-generated recommendation rationales, which will be fed into neural recommender systems. In this way, the insights from LLMs are effortlessly integrated into traditional recommender systems via a controllable reasoning pool.

We conduct extensive experiments on three datasets and six backbones to verify the improvement of ReLand on collaborative filtering and sequential recommendation. We also compared ReLand with several representative LLM-based methods [39, 44]. The results indicate that ReLand consistently improves the performance of traditional recommenders at a lower cost. Furthermore, we conduct online experiments with hundreds of millions of users, achieving statistically significant improvements of 3.19% in Click-Through Rate (CTR) and 1.08% in Conversion Rate (CVR). *Since January 2024, ReLand has been deployed in the recommendation system of Alipay, serving hundreds of millions of users daily.*

Overall, our contributions can be summarized as follows:

- To our knowledge, this is the first deployed LLM-enhanced recommendation framework tailored for industrial applications, capable of serving hundreds of millions of users.
- We propose ReLand, a novel LLM-enhanced recommendation framework, leveraging retrieval to effortlessly integrate large language models' insights into traditional recommender systems.
- Extensive offline and online experiments are conducted to verify the effectiveness and efficiency of ReLand.

2 PRELIMINARY

Encoding Items and Users. Formally, given the user set $\mathcal{U} = \{u\}$ and item set $\mathcal{I} = \{i\}$, the observed user-item interaction data is denoted as $\mathcal{D} = \{(u, i, t)\}$, where the instance (u, i, t) represents that user u interacts with item i at time t. In collaborative filtering, user interactions are viewed as an unordered set, while in sequential recommendation scenarios, they are organized into chronological sequences. In both scenarios, the item encoder transforms items with attributes $\mathcal{F} = \{f_i \mid i \in \mathcal{I}\}$ into dense vector representations:

$$\mathbf{E}_I = \text{ItemEncoder}\left(\mathcal{I}, \mathcal{F} \mid \Phi\right),$$

where $\mathbf{E}_I \in \mathbb{R}^{|\mathcal{I}| \times d}$ is the representations of items and Φ denotes the parameters of the item encoder. Generally, ItemEncoder$(\cdot)$ comprises an embedding layer for mapping item IDs and a text encoder

[1]Refer to the data reported in prior study [52], assuming 20 inferred tokens/user avg.

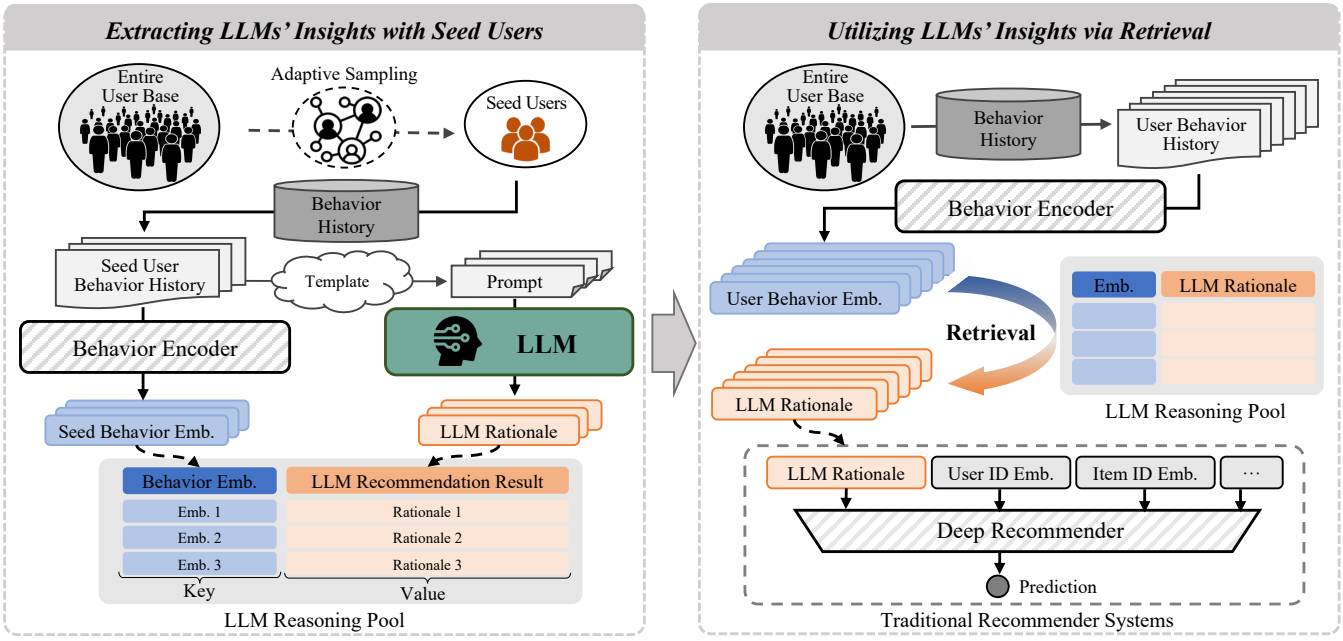


Figure 2: Illustration of ReLand. ReLand uses LLMs as recommenders to perform generative recommendations on seed users (left), then utilizes LLM-generated rationales to enhance the recommendation for the entire user base via retrieval (right).

(*e.g.*, BERT [4]) for processing item attributes (*e.g.*, title and category). Moreover, the behavior encoder generates dense representations for each user based on historical interactions:

$$\mathbf{E}_U = \text{BehaviorEncoder}\left(\mathcal{U}, \mathcal{D} \mid \Theta\right),$$

where $\mathbf{E}_U \in \mathbb{R}^{|\mathcal{U}| \times d}$ is the user representations and Θ denotes the parameters. In practice, the behavior encoder can be implemented with graph-based [10, 38] or sequence-based [13, 15] methods.

Prediction and Optimization. Given the above representations, we can calculate the user u's preference score $\hat{y}_{ui}$ for the item i with the dot product (or MLP layer followed by a sigmoid function [11]):

$$\hat{y}_{ui} = \boldsymbol{e}_i^\top \boldsymbol{e}_u,$$

where $\boldsymbol{e}_i$ and $\boldsymbol{e}_u$ are the representations of item i and user u, respectively. Finally, the model is optimized with BPR loss [28] or CE loss [15]. Despite their effectiveness, traditional methods are trained based on collaborative signals, lacking open-world knowledge.

3 METHODOLOGY

Next, we detail the LLM-based recommendation framework that leverages **Re**trieval to effortlessly integrate **La**rge language models' insights into i**nd**ustrial recommenders (short as **ReLand**).

Considering the intrinsic conflict in inference latency and resource cost between industrial recommender systems and LLMs, our framework seeks to minimize the deployment cost of LLMs by reducing the number of inferences, while maximizing the benefit of LLMs to recommender systems through knowledge reuse. Specifically, our proposed ReLand consists of two phases: 1) extracting LLMs' insights with seed users and 2) utilizing LLMs' insights via retrieval. The overview of ReLand is illustrated in Figure 2.

3.1 Extracting LLMs' Insights with Seed Users

As previously mentioned, when handling a user base of hundreds of millions, performing individual inferences for each user with LLMs is prohibitively expensive. To make industrial-scale systems benefit from LLMs' knowledge and insights, we propose carefully sampling a subset of users (referred to as "**seed users**") to extract high-quality, reusable recommendation results from LLMs, ensuring efficient use of computational resources. Building on previous works [14, 19], we refer to the reasoning texts generated by LLMs as "*rationales*" in the following sections, and specifically refer to those reasoning texts tailored recommendations as "*recommendation rationales*."

3.1.1 ***Adaptive Seed User Sampling Strategy.*** Before delving into the details of our sampling strategy, it is essential to clarify that the purpose of sampling seed users is to leverage them to extract high-quality and reusable reasoning knowledge from LLMs. Therefore, the seed users should mirror the preferences and behavior patterns of a broader user base. To achieve this objective, we develop an adaptive sampling strategy. Specifically, our sampling strategy merges representativeness-aware and importance-aware sampling to prioritize users who represent a broader user base or exhibit substantial interactive behavior:

- *Representativeness-aware Sampling*: The strategy is designed to capture the complex nature of user preferences through clustering and selecting users from each cluster. Specifically, we first employ a lightweight behavior encoder to transform the users' historical behavior into vector representations $\mathbf{M} \in \mathbb{R}^{|\mathcal{U}| \times d}$:

$$\mathbf{M} = \text{BehaviorEncoder}(\ \mathcal{U}, \mathcal{D}\ ;\ \Theta'), \tag{1}$$

where Θ' denotes the parameters of the lightweight behavior encoder, which can be implemented using various traditional

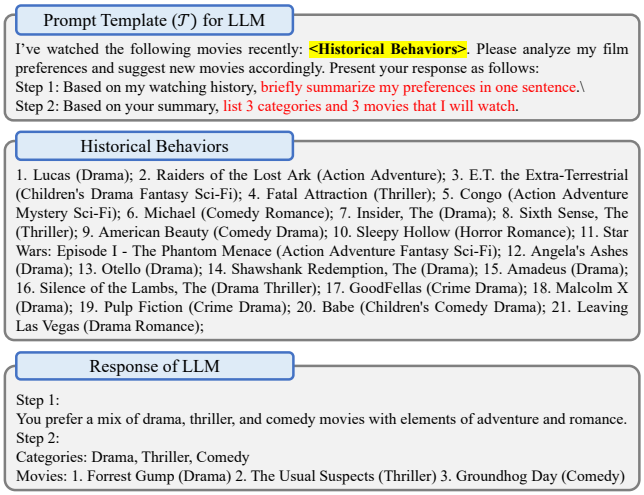


Figure 3: Step-by-step prompting for generating recommendation rationales (ChatGPT 3.5 is used as an example).

methods, such as LightGCN [10] and SASRec [15]. Next, we cluster the users into K clusters, and sample N_k users within the cluster C_k to form the representativeness-aware seed user set $\mathcal{S}_R$, where N_k is proportional to the size of the cluster $|C_k|$. In particular, if only one user is sampled in the cluster, we select the user closest to the cluster center as the seed user for that cluster. In this work, we employ K-means [24] as the cluster method. This strategy ensures that the selected users represent a broad user base, while also preserves the diversity among seed users.

- *Importance-aware Sampling*: For both traditional and LLM-based methods, sufficient interaction data is essential for accurate predictions. This is because the more interactions a user has contributed, the more reliable and consistent their information tends to be [23]. Therefore, this sampling strategy prioritizes users with more interactions. Formally, we define an indicator function $\mathbb{I}(u)$ for each sampling, whereby a user u is selected if $\mathbb{I}(u) = 1$. Hence, the importance-aware seed user set $\mathcal{S}_I$ can be defined as follows:

$$\mathcal{S}_I = \{u \mid u \in \mathcal{U}, \mathbb{I}(u) = 1\}, \ \mathbb{I}(u) \sim \text{Bernoulli}\left(\frac{|\mathcal{D}_u|}{\sum_{u \in \mathcal{U}} |\mathcal{D}_u|}\right), \quad (2)$$

where $\mathcal{D}_u$ denotes the interaction set of user u. By assigning higher sampling probabilities to users with more interactions, the importance-aware sampling strategy generates a more reliable and informative seed user set $\mathcal{S}_I$.

We alternately apply the two strategies mentioned above until we have sampled a predefined number of seed users to form the final seed user set $\mathcal{S} = \{u \mid u \in \mathcal{S}_R \cup \mathcal{S}_I\}$, where $|\mathcal{S}| \ll |\mathcal{U}|$. As a superior alternative to uniform sampling, the adaptive sampling strategy effectively balance representative users and important users, resulting in a high-quality seed user set $\mathcal{S}$.

3.1.2 ***Recommendation-oriented Prompting Strategy.*** Traditional recommender systems lack open-world knowledge and reasoning abilities, limiting both their accuracy and explainability. Significantly, LLMs present a substantial advancement in comprehending user behaviors [12, 44]. Thus, we guide LLMs via the Chain-of-Thought (CoT) prompting [41] to generate rationales for recommendations, including user preferences and favored categories/items. Specifically, our prompt template $\mathcal{T}$ are as follows:

- *Step 0 (optional for industry)* : Format user interaction data to produce a condensed interaction history.
- *Step 1* : Based on user interaction history, briefly summarize user preferences in one sentence.
- *Step 2*: Based on the summarized preferences, recommend categories and items to the user.

where "*Step 0*" can be optionally employed to filter out noisy text in industrial environment, such as promotional copy and other peripheral information. The CoT prompting guides LLMs to analyze user preferences step-by-step from a macro-to-micro perspective. As shown in Figure 3, LLMs can generate informative rationales for recommendations and infer categories/items that users might be interested in. With the template $\mathcal{T}$, we can construct prompts for each seed user and obtain the recommendation rationales $\mathcal{R} = \{r_s \mid s \in \mathcal{S}\}$ through the LLMs, where r_s denotes the rationale for the seed user s. Subsequently, the behavior representations and recommendation rationales of seed users form a controllable **LLM Reasoning Pool** $\mathcal{P} = \{(\boldsymbol{m}_s, r_s) \mid s \in \mathcal{S}, \boldsymbol{m}_s \in \mathbf{M}, r_s \in \mathcal{R}\}$, which contains a wealth of high-quality, reusable knowledge extracted from LLMs. By controlling the size and content of the LLM Reasoning Pool, we can balance the effectiveness and cost of the LLM-enhanced recommender (refer to Section 4.3.2 and Section 4.3.3).

3.2 Utilizing LLMs' Insights via Retrieval

By analyzing seed users, we have extracted LLMs' insights and constructed the LLM Reasoning Pool. However, applying the insights gained from small-scale seed users to the entire user base presents significant challenges. In this study, we assume that users with similar behaviors share the same recommendation rationales. Consequently, we propose scaling the insights from the LLM Reasoning Pool to the entire user base via retrieval, serving as supplementary information to enhance traditional recommender systems.

3.2.1 ***Scaling LLMs' Insights via Retrieval.*** To provide a reliable recommendation rationale for each user in the entire user base, we match each user with a seed user exhibiting the most similar behavioral pattern. We then retrieve the rationales of the seed users to serve as the rationale for their corresponding users. Specifically, we encode the historical behaviors of users into the representations matrix $\mathbf{M}$ with the lightweight behavior encoder (refer to Eq. (1)). Then, the recommendation rationale for user u can be retrieved as:

$$r_u = r_s, \quad s.t. \ s = \arg\max_{s \in \mathcal{S}} \frac{\boldsymbol{m}_u^\top \boldsymbol{m}_s}{||\boldsymbol{m}_u|| \cdot ||\boldsymbol{m}_s||}, \quad (3)$$

where $\boldsymbol{m}_u$ and $\boldsymbol{m}_s$ is the behavior representations of user u and seed user s. Since the representations encode users' historical behaviors, those with similar representations are likely to exhibit similar behavior patterns and have similar recommendation rationales.

3.2.2 ***Enhancing Recommender with LLMs' Insights.*** After assigning each user an LLM-generated rationale, we apply these rationales to enhance the traditional recommenders. Specifically, to effectively integrate the insights from LLMs with collaborative signals, we design a fusion layer to combine the rationale r_u with

ID embedding $\boldsymbol{e}_u$ in traditional recommender systems:

$$\boldsymbol{e}'_u = \text{FusionLayer}(\boldsymbol{e}_u,\ \text{TextEncoder}(r_u)), \tag{4}$$

where TextEncoder$(\cdot)$ converts the text of rationales into dense vectors and $\boldsymbol{e}'_u$ is the LLM-enhanced embedding of user u. Actually, ReLand does not limit the architecture of the text encoder and fusion layer. In our implementation, we employ BERT [4] to encode the text of Step 2 in rationales, while the text from Step 1 is considered to be an explanation for the recommendation. Besides, we implement the fusion layer with attention mechanisms [35] as:

$$\begin{aligned} \boldsymbol{z}_u &= \text{Linear}(\text{TextEncoder}(r_u)), \\ [\alpha_u^{(1)},\ \alpha_u^{(2)}] &= \text{softmax}\left([\boldsymbol{e}_u \cdot \mathbf{W},\ \boldsymbol{z}_u \cdot \mathbf{W}]\right), \\ \boldsymbol{e}'_u &= \alpha_u^{(1)} \boldsymbol{e}_u + \alpha_u^{(2)} \boldsymbol{z}_u, \end{aligned} \tag{5}$$

where Linear$(\cdot)$ transforms the text representations $\boldsymbol{z}_u$ to the same dimension with embedding $\boldsymbol{e}_u$ and $\mathbf{W} \in \mathbb{R}^d$ is a learnable attention matrix. In this manner, the open-world knowledge and insights from LLMs are adaptively integrated into user embeddings, thereby disrupting the closed recommender systems.

3.3 Comparison and Discussion

Generally, existing LLM-based recommendation methods can be classified into two categories: 1) LLMs as recommenders, and 2) LLM-enhanced recommenders. In contrast, ReLand first uses LLM as recommender to perform generative recommendations on seed users, then utilizes LLM-generated rationales to enhance the recommendation for the entire user base, which exhibits clear advantages.

Comparison with Methods Using LLMs as Recommenders. Such studies utilize LLMs to either directly generate recommendation lists [16–18, 49] or re-rank existing recommendation lists [3, 23, 46]. However, when LLMs are used as online recommenders to make real-time recommendations, the high cost and high response time of LLMs make it impractical in industrial settings. In contrast, ReLand only employs LLMs for offline inference, integrating LLMs' insights into industrial recommenders while adding almost no additional complexity for online serving.

Comparison with LLM-enhanced Recommenders. This line of work either treats LLMs as text encoders [22] to transfer descriptions of users and items into vectors, or leverages LLMs to infer reasoning information about items and users (e.g., the director of movie items or profile of users) [26, 39, 42, 44]. Although efficient, existing studies typically require individual inference for each user or items. For instance, the pioneering work KAR [44] applies LLMs to generate reasoning for each user and each item, while the latest SLIM [39] requires LLM inference for every interaction of each user. Thus, the offline complexities of KAR and SLIM can be approximated as $O(\beta \cdot (|\mathcal{U}| + |\mathcal{I}|))$ and $O(\beta \cdot |\mathcal{D}|)$, respectively, where β denotes the average complexity of LLMs per inference. Given LLMs' substantial costs (*i.e.,* β), this is still impractical in industrial settings with hundreds of millions of users and items. In contrast, ReLand only performs recommendations on small-scale seed users with LLMs and extends the LLM-generated rationales to the entire user base via retrieval. The offline complexity of ReLand can be approximated as $O(\beta|\mathcal{S}| + \gamma|\mathcal{U}|)$, where γ is the average complexity of retrieval for each user. Considering that $|\mathcal{S}| \ll |\mathcal{U}|$ and $\gamma \ll \beta$, the cost of ReLand is feasible for industrial deployment.

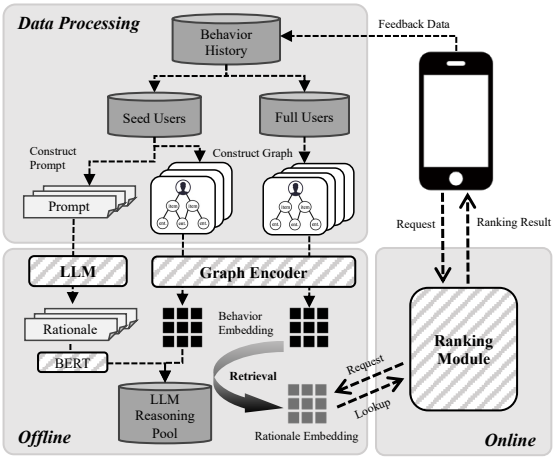


Figure 4: The deployment details of ReLand.

3.4 Advancing Towards Industrial Deployment

Currently, ReLand has been deployed on Alipay, serving hundreds of thousands of users daily. Considering the requirement for low-latency online inference, our framework is deployed in a hybrid offline-online manner, which can be summarized as: data processing → offline training → online serving.

The deployment details are shown in Figure 4. In particular, we adaptively sample approximately 75,000 seed users (out of a total user base of over 100 million) using the strategies in Sec. 3.1.1 and collect their behavioral data over the past 30 days. Then, we construct prompts and generate recommendation rationales for users using LLM, followed by a BERT encoder for representation learning. To mitigate potential privacy concerns, instead of directly accessing APIs of a black-box LLM (*e.g.,* ChatGPT 3.5 [2]), we opt to use the open-source LLM ChatGLM-6B [5], which has been fine-tuned with in-domain corpus from the online platform. As the success of ReLand heavily relies on accurately representing users based on their historical behaviors, we meticulously organize user behaviors across various scenarios into a unified graph to comprehensively and deeply understand user preference. Specifically, our deployed online platform, Alipay, offers a variety of services, such as E-commerce and short video, where we collect anonymized data on users' search, click, and transaction behaviors across the entire platform. Using this data, we established links between users and items, as well as items and entities, to create a heterogeneous graph where entities encompass brands, categories, topics, and more [47]. Subsequently, we utilize the unified graph to train a graph encoder using the classical GAT architecture [36], specifically tailored for link prediction[2]. Previously, we were able to generate detailed user representations with the well-trained graph encoder, as well as adequate recommendation rationales through the well-tuned LLM. These representations of behaviors and the

[2]The graph encoder's training pipeline won't burden our system as it's trained monthly.

Table 1: Statistics of the datasets.

Datasets	#Users	#Items	#Inters	#Inters/U.	Sparsity
Sports	35,598	18,357	296,337	8.3	99.94%
Beauty	22,363	12,101	198,502	8.9	99.93%
ML-1M	6,040	3,953	1,000,000	165.6	95.81%

aforementioned rationales for sampled seed users are then organized into the LLM reasoning pool. At last, for the entire user set, we utilize the vector retrieval engine Proxima[3] to efficiently match each user with a rationale embedding in an offline manner. Such an embedding matrix is then fed into the online ranking module as auxiliary features in both the training and inference stages.

Practical Analysis. Intuitively, the deployment of ReLand allows our online recommender to efficiently enjoy the superior reasoning capabilities of LLMs without adding significant burden to the offline training and online serving. Specifically, we present several practical cost analyses as follows: i) Generally, the reasoning pool is constructed weekly, and only 75,000 seed users are necessary for the LLM to reason, which usually takes approximately *three hours* with four NVIDIA A100 GPUs. Compared to the reasoning on the entire set (more than 100 million users, incurring over *six months*), this time investment is deemed acceptable and even negligible. ii) The rationale embedding matrix on the entire user set is updated daily. The generation process of behavior embeddings for all users is commonly implemented in a distributed way, which is completed in nine hours. And the vector retrieval engine Proxima is built upon a highly efficient indexing service, capable of retrieving the most relevant rationales for all users within two hours. iii) The rationale embedding imposes only a negligible burden on the online ranking module. Please refer to Section 5 for more details.

4 OFFLINE EXPERIMENTS

In this section, we conduct extensive offline experiments.

4.1 Experimental Settings

Dataset. We adopt three datasets: **Sports** and **Beauty**[4] correspond to the "Sports and Outdoors" and "Beauty" categories of the Amazon Dataset [9], respectively, where we utilize the title and brand as item features. **ML-1M**[5] contains a set of movie ratings [7], where we adopt the title and genres as movie features. Following previous works [10, 33, 45], we filter out users and items with fewer than 5 interactions. For the collaborative filtering task, we group interaction records by user and split them chronologically: 80% for training, 10% for validation, and 10% for testing. For the sequential recommendation task, we construct user interaction sequences by ordering interactions chronologically, capping the sequence length at 50. We apply a leave-one-out strategy: the last item in each user's sequence is for testing, the penultimate item for validation, and the rest for training. The details are summarized in Table 1.

[3]https://github.com/alibaba/proxima
[4]https://jmcauley.ucsd.edu/data/amazon/
[5]https://grouplens.org/datasets/movielens/1m/

Base Models. We select 6 methods from the two categories as base models, including *Collaborative Filtering* (*i.e.,* **NeuMF** [11], **NGCF** [38] and **LightGCN** [10]) and *Sequential Recommendation* (**GRU4Rec** [13], **SASRec** [15] and **DuoRec** [25]). Building on prior research [10, 21, 33, 45], we employ BPR loss [28] for all collaborative filtering methods and CE loss [15] for all sequential recommendation methods to ensure fair comparisons.

Baselines. We compare the proposed ReLand with two latest LLM-enhanced methods, *i.e.,* **KAR** [44] and **SLIM** [39]). Following the setting in the original paper, KAR and SLIM conduct the inference using LLMs for all users. However, it should be pointed out that this comparison is not completely fair for our approach since our method only infers using LLMs for a select 10% of users.

Implementation Details We implement our method and baseline methods with the open source recommendation framework `RecBole`[6] [50, 51]. All methods utilize the same dataloader, trainer, and evaluation modules, and their hyperparameters have been carefully tuned to ensure a fair comparison. For all methods, we utilize the Adam optimizer and implement an early stopping strategy with a patience setting of 10 epochs. Furthermore, we configure the embedding size to 64 and set the batch size at 1024. For ReLand, we adopt LightGCN [10] as the lightweight encoder (refer to Eq. (1)), and employed the K-means algorithm to cluster users into 1,780 clusters for the Sports dataset, 1,120 for the Beauty dataset, and 302 for the ML-1M dataset. We sample 10% of the entire user base as seed users. We employ ChatGPT-3.5 to generate recommendation rationales for seed users and made minor adjustments to the prompt for different datasets. Besides, we execute a full ranking evaluation and utilize top-K metrics, including Hit Ratio (HR@10) and Normalized Discounted Cumulative Gain (NDCG@10).

4.2 Overall Performance

As a model-agnostic framework, ReLand can significantly enhance existing recommendation methods, as showed in Table 2, where "$*$" denotes statistically significant improvement (p-value < 0.05) over the backbone. "ID-only" and "+ f_{item}" denote the base model and the extension with the item feature, where we concatenate the item ID embedding with the item feature representation to enhance the backbone. "ReLand" denotes the enhanced model that further introduces the LLM's knowledge based on "+ f_{item}".

Performance Gain for Collaborative Filtering. As illustrated in Table 2, ReLand improves all base models and outperforms their extensions with item features, demonstrating that ReLand offers valuable insights beyond item attributes. Furthermore, ReLand achieves performance gains on the GNN-based backbones (*i.e.,* NGCF and LightGCN), indicating that GNN's aggregated information cannot substitute for the reasoning knowledges of LLMs. Besides, ReLand exhibits greater improvement on the sparser datasets (*e.g.,* Sport and Beauty) compared to the denser ML-1M dataset, emphasizing LLMs' potential to mitigate the data sparsity issues.

Performance Gain for Sequential Recommendation. As Table 2 shows, our framework consistently improves performance

[6]https://github.com/RUCAIBox/RecBole

Table 2: Performance comparison *w.r.t.* different backbones enhanced by ReLand. Best result in bold; runner-up underlined.

Dataset	Metric	Collaborative Filtering Backbones								
		NeuMF			NGCF			LightGCN		
		ID-only	+ f_{item}	ReLand	ID-only	+ f_{item}	ReLand	ID-only	+ f_{item}	ReLand
Sports	NDCG@10	0.0088	0.0091	**0.0094***	0.0132	0.0134	**0.0136***	0.0155	0.0155	**0.0164***
	HR@10	0.0178	0.0189	**0.0195***	0.0272	0.0271	**0.0274**	0.0307	0.0308	**0.0322***
Beauty	NDCG@10	0.0154	0.0168	**0.0173***	0.0188	0.0193	**0.0200***	0.0216	0.0225	**0.0230***
	HR@10	0.0323	0.0351	**0.0360***	0.0388	**0.0419**	**0.0419**	0.0465	0.0470	**0.0476***
ML-1M	NDCG@10	0.0764	0.0767	**0.0797***	0.0808	0.0813	**0.0827***	0.0806	0.0794	**0.0818***
	HR@10	0.3570	0.3515	**0.3618***	0.3639	0.3671	**0.3684**	0.3659	0.3609	**0.3707***

Dataset	Metric	Sequential Recommendation Backbones								
		GRU4Rec			SASRec			DuoRec		
		ID-only	+ f_{item}	ReLand	ID-only	+ f_{item}	ReLand	ID-only	+ f_{item}	ReLand
Sports	NDCG@10	0.0183	0.0242	**0.0250***	0.0226	0.0243	**0.0257***	0.0241	0.0251	**0.0256***
	HR@10	0.0348	0.0474	**0.0475**	0.0484	0.0467	**0.0494***	0.0478	0.0485	**0.0488**
Beauty	NDCG@10	0.0343	**0.0406**	**0.0406**	0.0410	0.0429	**0.0440***	0.0426	0.0433	**0.0443***
	HR@10	0.0620	0.0724	**0.0776***	0.0831	0.0811	**0.0841***	0.0799	0.0811	**0.0829***
ML-1M	NDCG@10	0.1419	0.1451	**0.1474***	0.1167	0.1183	**0.1224***	0.1180	0.1196	**0.1245***
	HR@10	0.2551	0.2593	**0.2616***	0.2288	0.2253	**0.2303**	0.2276	0.2263	**0.2328***

Table 3: Comparison with LLMs-enhanced Methods. "# inference" denotes the number of LLM inferences required. "Time Cost" refers to the time required for inferring with LLMs.

Dataset	Metric	Base	KAR	SLIM	ReLand
Sports	NDCG@10	0.0226	0.0240	0.0234	**0.0257**
	HR@10	0.0484	0.0422	**0.0495**	0.0494
	# Inference	-	53,955	296,337	**3,556**
	Time Cost (h)	-	6.29	34.57	**0.41**
Beauty	NDCG@10	0.0410	0.0417	**0.0445**	0.0440
	HR@10	0.0831	0.0740	**0.0850**	0.0841
	# Inference	-	34,464	198,502	**2,236**
	Time Cost (h)	-	3.06	23.16	**0.26**
ML-1M	NDCG@10	0.1167	0.1209	0.1222	**0.1224**
	HR@10	0.2288	0.2263	0.2265	**0.2303**
	# Inference	-	9,993	1,000,0000	**604**
	Time Cost (h)	-	1.16	116.67	**0.07**

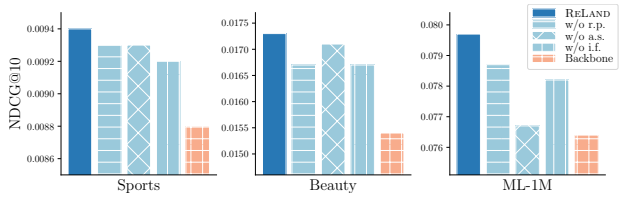
across three representative sequential backbones. Notably, incorporating item attributes into sequential methods yields a more substantial improvement than collaborative filtering, indicating the significance of item attributes in modeling sequential patterns. Furthermore, ReLand can enhance the performance of the Transformer-based methods (*i.e.,* SASRec and DuoRec), suggesting that LLMs' knowledge and reasoning capabilities can help attention mechanisms better capture complex patterns of user behaviors and are compatible with self-supervised methods (*i.e.,* DuoRec).

Superiority over LLM-enhanced Methods. As Table 3 shows, we further evaluated ReLand against the latest LLM-enhanced methods (*i.e.,* KAR and SLIM), where the base model is SASRec. It is important to note that this comparison is fundamentally unfair, as KAR and SLIM necessitate LLMs inference for the entire user base, while our method only infers for the selected 10% of users. From the table, we observe that KAR achieves a larger performance gain on the NDCG metric as compared to the HR metric, where NDCG is sensitive to the ranking order of recommendation result. This might suggest that the factual knowledge provided by KAR is more suitable for ranking tasks rather than recall tasks. Furthermore, the performances of SLIM and ReLand are comparable, while ReLand demonstrates a lower inference cost. As indicated by "# inference" in the table, we count the number of LLM inferences required by each method on each dataset. The amount of LLM inference required by ReLand is even less than 1% of that required by SLIM, yet ReLand achieves comparable performance gain.

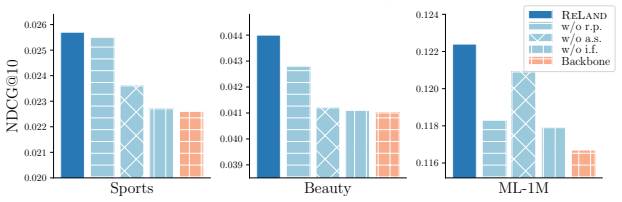
4.3 Further Analysis

In the following section, we provide a detailed analysis on ReLand.

4.3.1 ***Ablation Study.*** We perform an ablation study across three datasets for two scenarios. Here, we select NeuMF and SASRec as backbones, respectively. The experimental results are shown in Figure 5, which contains the backbone and three variants of ReLand : "*w/o* r.p." (replace recommendation-oriented prompting with directly prompting), "*w/o* a.s." (replace adaptive sampling with random sampling), "*w/o* i.f." (remove item feature embedding on the item side). The figure shows that when any module of ReLand is

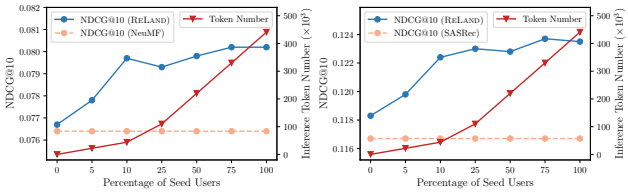


(a) Collaborative Filtering (NeuMF)



(b) Sequential Recommendation (SASRec)

Figure 5: Ablation study.

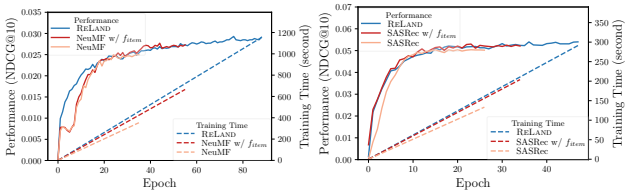


(a) Collaborative Filtering (b) Sequential Recommendation

Figure 6: Impact of the number of seed users on recommendation performance and inference token cost.

changed, the performance decreases significantly. Compared between the two scenarios, for sequential recommendation, performance gains depend more on adding item features on the item side. On the other hand, for collaborative filtering, utilizing LLM rationales alone (*i.e.,* "*w/o* i.f.") can also significantly enhance the backbone. Besides, the significant drop in ReLand's performance with random seed users (*i.e.,* "*w/o* a.s."), particularly leading to a negative impact when using NeuMF as the backbone of the ML dataset, indicates the critical importance of high-quality seed users for the framework's effectiveness.

4.3.2 ***Impact of the Number of Seed Users.*** ReLand achieves efficient LLM-based recommender systems by adaptively sampling seed users, where the number of seed users can balance the system's performance and efficiency. To further investigate the impact of the number of seed users, we adjust this number and analyze the performance in two scenarios. As illustrated in Figure 6, the model's performance improves with increased seed users, however, the marginal benefits diminish as the number increases further This indicates that increasing the number of seed users indeed improves the performance of our system. However, using LLMs to make individual inferences for the entire user base could potentially lead to resource wastage. Therefore, the number of seed users in industrial applications must be carefully determined.



(a) Collaborative Filtering (b) Sequential Recommendation

Figure 7: Curves of validation score on the Beauty dataset.

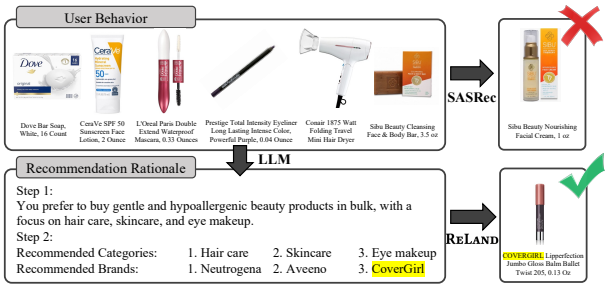


Figure 8: Case study. Comparison of the predictions for the next item obtained from SASRec and ReLand.

4.3.3 ***Cost Analysis of ReLand.*** Next, we analyze the costs of ReLand in terms of resources and time. As illustrated in Figure 6, by adjusting the proportion of seed users, we can effectively balance the trade-off between performance and resource consumption, where resource consumption is primarily measured by the number of tokens. In particular, sampling just 10% of users can achieve performance comparable to inferring for the entire user base, thereby significantly reducing resource costs. As shown in Figure 7, when running the same number of epochs, ReLand slightly outperforms the baseline on the validation set. Moreover, the training process of ReLand is more stable, and it ultimately achieves superior performance. Although our approach marginally increases the runtime per epoch, it is highly economical and efficient compared to existing methods that enhance recommender systems using LLMs.

4.3.4 ***Potential Interpretability of Recommendation.*** To illustrate the interpretability of recommendations introduced by ReLand, we present a case study in Figure 8 for a user with the ID "*A2LUSYXY8NXYKW*" from the Beauty dataset. In this case, although the *CoverGirl* brand had never appeared in the user's historical behavior, ReLand still successfully recommends the target item of the *CoverGirl* brand. In contrast, SASRec recommends another item from the same *Sibu* brand as the item with which the user last interacted. Besides, the LLM accurately summarizes the user's preferences in Step 1, including specific needs for "gentle and hypoallergenic," and "buy in bulk." This indicates that ReLand can provide a natural language explanation, enhancing the explainability of recommendations.

5 ONLINE EXPERIMENT

To evaluate the performance of ReLand in industrial environment, we randomly serve 10% of users with our method and another

Table 4: Overall online improvement results of ReLand. "$*$" denotes the statistical significance at the 95% level.

Days	1	2	3	4	5	Overall
CTR	*3.61%	*4.04%	*3.05%	*2.58%	*2.83%	*3.19%
CVR	-0.31%	0.37%	0.52%	*1.29%	*1.51%	*1.08%

Table 5: Online improvement results on different active-level users. The activity gradually increases from $L1$ to $L4$.

Activity Level	$L1$	$L2$	$L3$	$L4$
CTR	4.02%	2.07%	4.69%	3.06%
CVR	10.08%	3.73%	1.89%	-0.39%

10% of users with the baseline, which is built upon MaskNet [40], with input features comprising user/item profiles, user behavior sequences, and so on. We conduct a five-day A/B test from 2024-01-14 to 2024-01-18, impacting approximately 2.7×10^7 users.

Overall Performance. We employ Click-Through Rate (CTR) and Conversion Rate (CVR) to evaluate the online performance. Both models are trained with the same dataset and deployed on the same server cluster to ensure fairness. Table 4 shows the results of the enhanced model with our method compared to the baseline model during the test period. We could see that our model consistently outperform baseline during the A/B test and overall achieved a statistically significant improvement of 3.19% and 1.08% on CTR and CVR, respectively. Currently, our framework has been deployed and served hundreds of millions of users of Alipay [7].

Segmented Population Analysis. In real-world recommender systems, low-frequency users often receive suboptimal recommendation results. To validate the impact of our method on various segmented populations, we categorize users into four activity levels (*e.g.,* $L1 \sim L4$) based on their click and transaction frequencies, where $L4$ represents the most active user segment, while $L1$ denotes the least active one. As shown in Table 5, our framework improves recommendation performance for low-frequency users (*e.g.,* $L1$) significantly more than for high-frequency users (*e.g.,* $L4$). This indicates that the LLMs' insight can effectively complement user behavior sparsity in industrial recommenders.

6 RELATED WORK

In the literature on recommender systems [30], collaborative filtering (CF) [10, 11, 27, 28, 38] and sequential recommendation [13, 15, 25, 32, 37, 45] remain the fundamental techniques to capture users' preferences. However, these traditional recommender systems, typically trained and deployed in isolated environments, inevitably encounter isolation issues [44]. Inspired by the success of Large Language Models (LLMs) [52], recent studies [6, 43, 44, 48] have attempted to leverage the open-world knowledge of LLMs to enhance recommendation. For example, TALLRec [1], RecRanker [23] and LLaRA [18] directly treat LLMs as recommenders to generate recommendation lists that satisfy user interests. Subsequently, POD [16], E4SRec [17], CoLLM [49], and CLLM4Rec [53] propose incorporating collaborative information into LLMs to improve recommendation performance. However, the high response time of LLMs makes these methods impractical for real-time recommendation in industrial settings. Therefore, KAR [44], LLMRec [42] and other pioneering works [22, 26] leverage the reasoning capabilities of LLMs to obtain factual information for items and users offline, without adding significant burden to the online serving. More recently, SLIM [39] leverages LLMs in a resource-efficient manner with the help of knowledge distillation. However, these works require individual inference for each user or item using LLMs, leading to unaffordable computational costs. In the course of our study, we noticed a contemporaneous work called LLM-CF [31], which is similar to our framework in terms of retrieval. However, LLM-CF relies on a cumbersome transformer to generate features for the pairs of users and candidate items, which limits its applicability in large-scale industrial recommenders. Overall, although effective, none of the above works can be integrated into industry recommenders with hundreds of millions of users.

7 CONCLUSIONS

In this work, we present an LLM-enhanced recommendation framework (**ReLand**), which leverages **Re**trieval to integrate **La**rge language models' insights into i**nd**ustrial recommenders at an affordable cost. Specifically, we adaptively sample seed users and then employ LLMs to generate recommendation rationales for them, which are used to construct an LLM Reasoning Pool. Subsequently, we retrieve reliable rationales from the LLM Reasoning Pool and apply them as auxiliary features across the entire user base, thereby effectively enhancing the industrial recommender system. Extensive offline and online experiments confirm the efficacy of ReLand. Currently, ReLand has been deployed in the recommender system of Alipay, serving hundreds of millions of users daily.

Limitations and Future Work. We bring to light the prospect of LLMs to empower industrial recommender systems while identifying persisting limitations for advancing future research. i) *Diversity of recommendation rationales.* To attain higher efficiency, ReLand tends to retrieve identical recommendation rationales for several users, thereby diminishing the diversity of rationales among the user base. Going forward, retrieving multiple rationales for each user and integrating them may enhance the diversity of rationales. ii) *Optimal number of seed users.* As the number of seed users increases, the recommendation performance does not grow linearly (detailed in Section 4.3.2). In practice, we balance computing power with the number of seed users through careful tuning. Nevertheless, efficiently determining the optimal number of seed users to maximize the Return on Investment (ROI) remains a compelling issue for exploration. iii) *User cold start problem.* Within our framework, LLMs leverage users' historical behaviors along with open-world knowledge to infer user preferences. However, it struggles with new users without any previous interactions. Therefore, future improvements could include using user profiles (*e.g.,* age, gender, and occupation) or assigning specific seed users to new users for better "cold start" recommendations.

[7] After full deployment, an additional 14-day comparative tracking experiment showed our model achieved a 1.53% increase in CTR and a 1.02% increase in CVR compared to the original baseline, confirming its sustained effectiveness.

REFERENCES

[1] 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*. 1007–1014.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems* 33 (2020), 1877–1901.

[3] Zheng Chen. 2023. PALR: Personalization Aware LLMs for Recommendation. *arXiv preprint arXiv:2305.07622* (2023).

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[5] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 320–335.

[6] 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 Sixteenth ACM Conference on Recommender Systems*.

[7] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems* 5, 4 (2015).

[8] Rachel M Harrison, Anton Dereventsov, and Anton Bibin. 2023. Zero-shot recommendations with pre-trained large language models for multimodal nudging. In *2023 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 1535–1542.

[9] 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*. 507–517.

[10] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 639–648.

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.

[12] Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. 2023. Large language models as zero-shot conversational recommenders. In *CIKM*. 720–730.

[13] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations*.

[14] Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*. 8003–8017.

[15] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *International Conference on Data Mining*. 197–206.

[16] 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*. 1348–1357.

[17] Xinhang Li, Chong Chen, Xiangyu Zhao, Yong Zhang, and Chunxiao Xing. 2023. E4SRec: An elegant effective efficient extensible solution of large language models for sequential recommendation. *arXiv preprint arXiv:2312.02443* (2023).

[18] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. Llara: Large language-recommendation assistant. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1785–1795.

[19] Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Hao Zhang, Yong Liu, Chuhan Wu, Xiangyang Li, Chenxu Zhu, Huifeng Guo, Yong Yu, Ruiming Tang, and Weinan Zhang. 2024. How Can Recommender Systems Benefit from Large Language Models: A Survey. *ACM Trans. Inf. Syst.* (jul 2024).

[20] Jianghao Lin, Rong Shan, Chenxu Zhu, Kounianhua Du, Bo Chen, Shigang Quan, Ruiming Tang, Yong Yu, and Weinan Zhang. 2024. Rella: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation. In *Proceedings of the ACM on Web Conference 2024*. 3497–3508.

[21] Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving Graph Collaborative Filtering with Neighborhood-enriched Contrastive Learning. In *WWW*. 2320–2329.

[22] 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 Seventeen ACM International Conference on Web Search and Data Mining*.

[23] 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* (2023).

[24] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1. 281–297.

[25] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2022. Contrastive learning for representation degeneration problem in sequential recommendation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 813–823.

[26] Xubin Ren, Wei Wei, Lianghao Xia, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Representation learning with large language models for recommendation. In *Proceedings of the ACM on Web Conference 2024*. 3464–3475.

[27] Steffen Rendle. 2010. Factorization machines. In *The 10th IEEE International Conference on Data Mining*. 995–1000.

[28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. 452–461.

[29] Scott Sanner, Krisztian Balog, Filip Radlinski, Ben Wedin, and Lucas Dixon. 2023. Large language models are competitive near cold-start recommenders for language-and item-based preferences. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 890–896.

[30] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* 2009 (2009).

[31] Zhongxiang Sun, Zihua Si, Xiaoxue Zang, Kai Zheng, Yang Song, Xiao Zhang, and Jun Xu. 2024. Large Language Models Enhanced Collaborative Filtering. arXiv:2403.17688 [cs.IR]

[32] 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*. 565–573.

[33] Changxin Tian, Binbin Hu, Wayne Xin Zhao, Zhiqiang Zhang, and Jun Zhou. 2023. Periodicity May Be Emanative: Hierarchical Contrastive Learning for Sequential Recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 2442–2451.

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

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 30 (2017).

[36] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

[37] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. 6332–6338.

[38] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 165–174.

[39] Yuling Wang, Changxin Tian, Binbin Hu, Yanhua Yu, Ziqi Liu, Zhiqiang Zhang, Jun Zhou, Liang Pang, and Xiao Wang. 2024. Can Small Language Models be Good Reasoners for Sequential Recommendation?. In *Proceedings of the ACM on Web Conference 2024*. 3876–3887.

[40] Zhiqiang Wang, Qingyun She, and Junlin Zhang. 2021. MaskNet: Introducing Feature-Wise Multiplication to CTR Ranking Models by Instance-Guided Mask. *arXiv preprint arXiv:2102.07619* (2021).

[41] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[42] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 806–815.

[43] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, and Enhong Chen. 2023. A Survey on Large Language Models for Recommendation. arXiv:2305.19860 [cs.IR]

[44] Yunjia Xi, Weiwen Liu, Jianghao Lin, Jieming Zhu, Bo Chen, Ruiming Tang, Weinan Zhang, Rui Zhang, and Yong Yu. 2023. Towards Open-World Recommendation with Knowledge Augmentation from Large Language Models. *arXiv preprint arXiv:2306.10933* (2023).

[45] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive learning for sequential recommendation. In *38th IEEE International Conference on Data Engineering*. 1259–1273.

[46] 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* (2023).

[47] Xiaoling Zang, Binbin Hu, Jun Chu, Zhiqiang Zhang, Guannan Zhang, Jun Zhou, and Wenliang Zhong. 2023. Commonsense Knowledge Graph towards Super APP and Its Applications in Alipay. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5509–5519.

[48] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *arXiv preprint arXiv:2305.07001* (2023).

[49] Yang Zhang, Fuli Feng, Jizhi Zhang, Keqin Bao, Qifan Wang, and Xiangnan He. 2023. Collm: Integrating collaborative embeddings into large language models for recommendation. *arXiv preprint arXiv:2310.19488* (2023).

[50] Wayne Xin Zhao, Yupeng Hou, Xingyu Pan, Chen Yang, Zeyu Zhang, Zihan Lin, Jingsen Zhang, Shuqing Bian, Jiakai Tang, Wenqi Sun, et al. 2022. RecBole 2.0: Towards a More Up-to-Date Recommendation Library. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 4722–4726.

[51] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, et al. 2021. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4653–4664.

[52] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).

[53] Yaochen Zhu, Liang Wu, Qi Guo, Liangjie Hong, and Jundong Li. 2024. Collaborative large language model for recommender systems. In *Proceedings of the ACM on Web Conference 2024*. 3162–3172.