RecLM: Recommendation Instruction Tuning

Yangqin Jiang
University of Hong Kong
Hong Kong, China
mrjiangyq99@gmail.com

Yuhao Yang
University of Hong Kong
Hong Kong, China
yuhao-yang@outlook.com

Lianghao Xia
University of Hong Kong
Hong Kong, China
aka_xia@foxmail.com

Da Luo
Wechat, Tencent
Guang Zhou, China
lodaluo@tencent.com

Kangyi Lin
Wechat, Tencent
Guang Zhou, China
plancklin@tencent.com

Chao Huang[*]
University of Hong Kong
Hong Kong, China
chaohuang75@gmail.com

Abstract

Modern recommender systems aim to deeply understand users' complex preferences through their past interactions. While deep collaborative filtering approaches using Graph Neural Networks (GNNs) excel at capturing user-item relationships, their effectiveness is limited when handling sparse data or zero-shot scenarios, primarily due to constraints in ID-based embedding functions. To address these challenges, we propose a model-agnostic recommendation instruction-tuning paradigm that seamlessly integrates large language models with collaborative filtering. Our proposed Recommendation Language Model (RecLM) enhances the capture of user preference diversity through a carefully designed reinforcement learning reward function that facilitates self-augmentation of language models. Comprehensive evaluations demonstrate significant advantages of our approach across various settings, and its plug-and-play compatibility with state-of-the-art recommender systems results in notable performance enhancements. We have made our RecLM available anonymously at: https://github.com/HKUDS/RecLM.

1 Introduction

Recommendation systems serve as essential components of modern web applications, helping users navigate through the vast digital information landscape. These systems provide personalized suggestions across diverse platforms, including product recommendations on e-commerce platforms (Wang et al., 2020; Wu et al., 2018), content discovery on social networking sites (Jamali and Ester, 2010; Zhang et al., 2021), and viewer-tailored suggestions on video sharing platforms (Zhan et al., 2022; Jiang et al., 2024). At the core of these systems lies Collaborative Filtering (CF), a widely adopted approach that harnesses collective user preferences to generate personalized recommendations.

[*]Chao Huang is the corresponding author.

Current recommender systems predominantly operate within the user/item ID paradigm, where training data consists primarily of mapped user and item indices. While this approach has driven significant advancements in recommendation technology, particularly in data-rich scenarios (Yao et al., 2021; Yuan et al., 2023), it faces several fundamental limitations. Most notably, these systems encounter substantial challenges in cold-start scenarios and struggle to generalize effectively in zero-shot learning situations. The inherent dependency on ID-based representations becomes particularly problematic in completely cold-start settings, where systems fail to generate meaningful representations for new items, ultimately compromising their ability to deliver accurate recommendations.

Addressing the fundamental cold-start challenge in ID-based recommendation systems requires moving beyond traditional ID-based embeddings to leverage external features (*e.g.*, textual or visual information) for generating user and item representations. However, this promising approach encounters significant practical challenges in real-world applications, primarily stemming from two critical issues: data incompleteness and quality concerns. On the incompleteness front, users frequently withhold personal information due to privacy concerns, leading to substantial gaps in external features. Simultaneously, existing data often suffers from quality issues, manifesting as misleading tags, irrelevant product specifications, or unreliable item descriptions. These challenges collectively underscore the critical need for robust methods to extract accurate, relevant, and high-quality features from inherently noisy and incomplete side information—a prerequisite for achieving effective recommendation generalization in data-scarce scenarios.

The exceptional generalization and reasoning capabilities of Large Language Models (LLMs) motivate a novel solution to cold-start recommendation challenges through specialized profiling systems based on external side information. This approach centers on developing effective language models customized for recommendation tasks, addressing two critical challenges: (i) designing robust mechanisms for LLMs to generate accurate profile representations that capture essential recommendation characteristics, especially for users or items lacking historical interactions, and (ii) developing techniques for LLMs to distill high-quality profiles from noisy feature sets while preserving the integrity of user-item interaction patterns and behavioral context. While recent LLM-based recommenders leverage LLMs to enhance the performance of recommenders (Chen et al., 2024; Zhang et al., 2024; Lyu et al., 2023), they do not specifically address the challenges associated with cold-start scenarios and often concentrate solely on learning the preferences of individual users or improving the features of individual items, thereby neglecting the collaborative relationships inherent in traditional CF recommenders.

Contribution. To address these challenges, we propose RecLM, a novel recommendation instruction tuning paradigm that revolutionizes how LLMs understand behavioral contexts in recommender systems. While LLMs demonstrate superior natural language processing capabilities, they fundamentally lack the ability to model complex user-item interactions and behavioral preferences. Our approach tackles this limitation through two key technical innovations: (1) a seamless integration mechanism that fuses external user-item features with collaborative interaction patterns through specialized instruction tuning, enabling effective cold-start profiling without direct supervision signals, and (2) a reinforcement learning-based personalized feature enhancement framework that systematically reduces noise and mitigates over-smoothing effects inherent in collaborative filtering. This model-agnostic framework can be plugged into existing recommenders to significantly enhance their generalization capacity, particularly in data-scarce scenarios. Our main contributions are as follows:

- **Model-Agnostic Framework**. We introduce a model-agnostic instruction tuning framework RecLM. It can be seamlessly integrated into existing recommender systems as a plug-and-play component, significantly enhancing their generalization capacity in scenarios with data scarcity.

- **Enhancing Profiling System**. We integrate large language models with collaborative filtering to enhance user profiling and representation, particularly in cold-start scenarios, where current methods often struggle. Additionally, our approach employs reinforcement learning to refine profile quality, effectively addressing challenges associated with data noise and over-smoothing.

- **Comprehensive Evaluation**. We integrate RecLM with several cutting-edge recommenders to assess the effectiveness of our method in various settings. This involves performing ablation studies and efficiency evaluations. Moreover, we conduct comprehensive experiments in real-world industrial recommendation scenarios, showcasing the practicality and scalability of RecLM.

2 Related Work

2.1 ID-based Recommender Systems

In recommender systems, numerous collaborative filtering (CF) models have been proposed to map users and items into latent representations based on user/item IDs (Koren et al., 2021; Su and Khoshgoftaar, 2009). These methods have evolved significantly, from early matrix factorization techniques, such as BiasMF (Koren et al., 2009), to the introduction of Neural Collaborative Filtering with the advent of neural networks (He et al., 2017). Recently, advances in graph neural networks (GNNs) have opened promising avenues for constructing bipartite graphs based on user-item interaction history, allowing for the capture of high-order collaborative relationships. GNN-based methods, including NGCF (Wang et al., 2019), GCCF (Chen et al., 2020), and LightGCN (He et al., 2020), have demonstrated the performance of SOTA, increasing the effectiveness of the recommendation.

In addition, researchers have incorporated self-supervised learning (SSL) techniques as supplementary learning objectives to improve the robustness of recommenders and address challenges related to data sparsity and noise (Yu et al., 2023). Contrastive learning (CL), a widely adopted SSL technique, has been effectively applied in CF research through approaches such as SGL (Wu et al., 2021), SimGCL (Yu et al., 2022), NCL (Lin et al., 2022), and AdaGCL (Jiang et al., 2023). Despite these advancements, ID-based recommenders still face significant limitations, particularly in completely cold-start scenarios and in terms of model transferability (Yuan et al., 2023).

2.2 LLMs for Recommendation

The integration of large language models (LLMs) into recommender systems has attracted considerable interest (Fan et al., 2023; Lin et al., 2023; Liu et al., 2023). Existing approaches fall into two primary categories. The first category, including methods like P5 (Geng et al., 2022), LLM-Rec (Lyu et al., 2023), Chat-REC (Gao et al., 2023), and ReCRanker (Luo et al., 2024), focuses on designing task-aligned prompts or leveraging LLMs' reasoning capabilities, using LLMs directly as inference models. The second category enhances traditional collaborative filtering (CF) methods by integrating LLMs. For example, LEARN (Jia et al., 2024) aligns LLM semantic spaces with recommendation objectives via contrastive learning, HLLM (Chen et al., 2024) employs hierarchical LLM architectures for multi-granularity recommendations, LLM-Rec (Wei et al., 2024) augments user-item interaction graphs using LLMs, and RLMRec (Ren et al., 2023) combines LLM-enhanced text embeddings with GNN-based representations. However, these methods often lack task-specific fine-tuning and primarily target full-shot scenarios.

In contrast, our work proposes a novel instruction-tuning technique for open-source LLMs, enabling adaptation to specific recommendation tasks and effective collaborative information capture for profile generation. While approaches like InstructRec (Zhang et al., 2023) and TALLRec (Bao et al., 2023) align LLMs with recommendation tasks, they face scalability issues due to instruction-question-answering prompts and exhibit limited generalization on sparse data. Our method improves generalization in data-scarce and noisy environments while maintaining efficiency for large-scale practical applications.

3 Preliminaries

ID-based Collaborative Filtering. In the ID-based collaborative filtering (CF) paradigm, the primary goal is to optimize the ID embeddings of users and items. This optimization aims to accurately capture and represent user preferences for items, while considering the interaction patterns of users and items that are similar. Formally, we have a set of users denoted as $\mathcal{U} = \{u_1, \cdots, u_I\}$, and a set of items denoted as $\mathcal{V} = \{v_1, \cdots, v_J\}$. Each user and item is assigned initial ID embeddings, represented as $\mathbf{x}_u$ and $\mathbf{x}_v \in \mathbb{R}^d$ respectively. The objective is to obtain optimized user and item representations, denoted as $\mathbf{e}_u, \mathbf{e}_v \in \mathbb{R}^d$, through a recommender model $\mathcal{R}(\mathbf{x}_u, \mathbf{x}_v)$. This model aims to maximize the posterior distribution $p(\mathbf{e}|\mathcal{X}) \propto p(\mathcal{X}|\mathbf{e})p(\mathbf{e})$. The predicted likelihood of user-item interaction, denoted as $\hat{y}_{u,v}$, is derived by performing a dot product between the user and item representations, as follows: $\hat{y}_{u,v} = \mathbf{e}_u^\top \cdot \mathbf{e}_v$.

While state-of-the-art (SOTA) recommenders operating within the ID-based collaborative filtering paradigm have exhibited remarkable performance, they face significant challenges when tasked with handling cold-start recommendation scenarios, especially in situations where data scarcity is prevalent. The primary obstacle that ID-based recommenders encounter in these situations stems from the lack of past interaction history for newly introduced items. This absence of accumulated user engagement data disrupts the optimization paradigm that these systems typically rely upon, thereby making it considerably more difficult to generate accurate and meaningful representations for these items. Consequently, ID-based recommenders may encounter difficulties in effectively modeling and understanding the inherent characteristics and preferences associated with these cold-start items, leading to a notable decline in the overall performance, particularly in zero-shot scenarios where no prior interaction data is available for certain items.

4 Methodology

4.1 Text-Enhanced Representations

To address the challenge of cold-start items in zero-shot recommendation scenarios, we propose a novel approach that leverages textual side features for generalized and adaptable user and item representation learning. Specifically, we seek to replace the traditional ID-based embeddings with the side information associated with the items, namely their text descriptions, represented as $\mathbf{F} \in \mathbb{R}^{|\mathcal{V}| \times d_t}$. To accomplish this, we utilize a project layer with a multi-layer perceptron T_{raw} to map the raw textual features $\mathbf{f} \in \mathbb{R}^{d_t}$ into a lower-dimensional latent space $\mathbb{R}^d$. The resulting representation $\hat{\mathbf{f}} \in \mathbb{R}^d$ is then used as the initial item representation:

$$\hat{\mathbf{f}}_v = T_{raw}(\mathbf{f}). \tag{1}$$

This text-driven method seamlessly combines collaborative signals with textual semantics, allowing our recommender to accurately capture user preferences. By using rich textual features as item representations, we optimize user ID embeddings

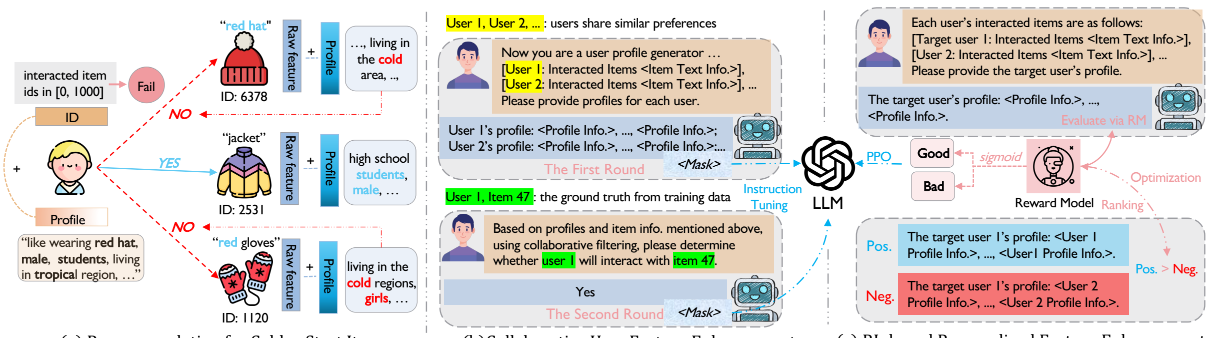

"red hat"
interacted item ids in [0, 1000]
Fail
ID
NO
YES
NO
Profile
"like wearing red hat, male, students, living in tropical region, ..."
ID: 6378
"jacket"
ID: 2531
"red gloves"
ID: 1120
Raw feature
Profile
..., living in the cold area, ...
high school students, male, ...
living in the cold regions, girls, ...
User 1, User 2, ... : users share similar preferences
Now you are a user profile generator ...
[User 1: Interacted Items <Item Text Info.>],
[User 2: Interacted Items <Item Text Info.>], ...
Please provide profiles for each user.
User 1's profile: <Profile Info.>, ..., <Profile Info.>;
User 2's profile: <Profile Info.>, ..., <Profile Info.>;...
The First Round
<Mask>
User 1, Item 47 : the ground truth from training data
Based on profiles and item info. mentioned above, using collaborative filtering, please determine whether user 1 will interact with item 47.
Yes
The Second Round
<Mask>
Instruction Tuning
LLM
PPO
Each user's interacted items are as follows:
[Target user 1: Interacted Items <Item Text Info.>],
[User 2: Interacted Items <Item Text Info.>], ...
Please provide the target user's profile.
The target user's profile: <Profile Info.>, ..., <Profile Info.>.
Evaluate via RM
Good
Bad
sigmoid
Reward Model
Optimization
Ranking
Pos.
The target user 1's profile: <User 1 Profile Info.>, ..., <User1 Profile Info.>.
Neg.
The target user 1's profile: <User 2 Profile Info.>, ..., <User 2 Profile Info.>.
Pos. > Neg.
(a) Recommendation for Cold − Start Items (b)Collaborative User Feature Enhancement (c) RL-based Personalized Feature Enhancement

Figure 1: Overall framework of the proposed RecLM.

based on interactions, infusing valuable collaborative cues into user and item representations to adapt to changing preferences. This enables our model to understand users' affinities for text-based items and provides zero-shot prediction capabilities for cold-start items with no prior interaction data.

LLM-enhanced User/Item Profiling. To further empower user representations with the rich textual semantics provided by large language models (LLMs), we propose generating user profile information that can reflect their interaction preferences. Specifically, item profiles can be derived from the profiles of users who frequently interact with them. This approach proves valuable in capturing user preferences and facilitating accurate recommendations for cold-start items. On the user side, the ID embedding $\mathbf{x}_u \in \mathbb{R}^d$ integrates with the user profile $\mathbf{p}_u \in \mathbb{R}^{d_t}$, allowing the system to leverage the user's ID-based embedding and their generated profile, which can capture more nuanced preferences. On the item side, the raw text $\mathbf{x}_v$ combines with the item profile $\mathbf{p}_v \in \mathbb{R}^{d_t}$, enabling the system to better understand the item's characteristics and how they align with user preferences.

$$\hat{\mathbf{f}}_u^{aug} = \Psi(\mathbf{x}_u \mid T_{pro}(\mathbf{p}_u)), \\ \hat{\mathbf{f}}_v^{aug} = \Psi(\hat{\mathbf{f}}_v \mid T_{pro}(\mathbf{p}_v)). \tag{2}$$

Our framework employs a dual-MLP architecture for effective multi-modal fusion: an initial MLP encoder $\Psi(\cdot)$ consolidates heterogeneous features, followed by a profile transformation network T_{pro} that projects the unified embeddings into the model's latent space. This hierarchical process generates enriched user and item representations $\hat{\mathbf{f}}_u^{aug} \in \mathbb{R}^d$ and $\hat{\mathbf{f}}_v^{aug} \in \mathbb{R}^d$, capturing comprehensive behavioral patterns. To further enhance representation quality, we leverage SOTA LLMs as profile augmentation engines, generating supplementary semantic information that significantly boosts our recommender system's modeling capacity.

4.2 Enhancing Collaborative Features via Recommendation Instruction Tuning

Our RecLM framework aims to align collaborative relationships with textual semantic representations through an innovative recommendation instruction tuning paradigm that integrates user collaborations into the LLM-based profiling process. It employs a two-phase approach: first, it uses knowledge distillation and dialogue-based instruction tuning to maintain high-order collaborative similarities, creating informative user profiles. Then, it utilizes these refined user profiles to generate semantically aligned item profiles, resulting in a cohesive representation space that effectively captures both individual characteristics and collaborative patterns.

4.2.1 LLM Tuning with Collaborative Signals

Our approach utilizes open-source LLMs, such as llama2-7b-chat and llama3.1-8b-instruct, as the foundation, enhanced by a novel collaborative instruction tuning framework that captures both user-item interactions and profile generation capabilities. We specifically design specialized prompt templates that incorporate collaborative signals and create input-output pairs using ChatGPT to identify various user-item relationship patterns. In this context, ChatGPT serves as a powerful LLM with strong text summarization abilities, generating text profiles that characterize interaction patterns among users and items, as well as their collaborative relationships, ultimately resulting in textual instructions for collaborative instruction tuning.

4.2.2 Collaborative Instruction Tuning

In our recommendation instruction tuning paradigm, we introduce a two-turn collaborative instruction tuning approach to inject higher-order collaborative relational context. By incorporating collaborative signals that capture complex user-user and user-item relationships, this method overcomes the limitations of those relying only on direct interactions. Our framework empowers LLMs to create richer user profiles by leveraging collaborative network dynamics, while also tackling the fundamental challenge of insufficient direct supervision in profile generation tasks.

Profile Generation with Two-turn Dialogue. A fundamental challenge in LLM-based profile generation lies in the absence of ground truth data for direct evaluation, forcing practitioners to rely on indirect assessment through downstream recommendation performance. To address this limitation, we propose a novel two-turn collaborative instruction tuning paradigm that provides explicit supervision signals. Unlike conventional approaches that struggle with profile quality assessment, our framework leverages collaborative user relationships to guide LLMs in generating high-quality profiles through structured dialogue interactions.

• **First Turn - Collaborative Profile Generation:** The initial turn takes input query $\mathcal{Q}$ containing both the target user's historical item interactions and those of similar users identified from their collaborative neighborhood. Here, user similarities are measured by the encoded user embeddings from LightGCN model (He et al., 2020). The LLM generates output response $\mathcal{R}$ consisting of user profiles that capture collaborative patterns.

$$\begin{aligned}\mathcal{Q}_{fir.} &= Prompt(u_t, \{u_n\}, \mathcal{V}_t, \{\mathcal{V}_n\}),\\ \mathcal{R}_{fir.} &= Prompt(u_t, \{u_n\}, \mathcal{P}_t, \{\mathcal{P}_n\}).\end{aligned} \tag{3}$$

• **Second Turn - Supervised Interaction Prediction:** The second turn reformulates the instruction-tuning task as an interaction prediction problem, aiming to enhance profile generation by incorporating explicit user-item preference signals. Specifically, the input query $\mathcal{Q}$ prompts whether target user u_t will interact with candidate item v_t, while the output response $\mathcal{R}$ corresponds to the ground truth interaction status (*i.e.*, yes or no) from the training dataset. To ensure balanced and informative training signals, we employ a systematic sampling strategy: For positive samples (50%), we select v^+ from the user's interaction history $\mathcal{V}_t$ under the constraint that v^+ also appears in similar users' history $\mathcal{V}_n$, while removing it from $\mathcal{V}_t$ in first-turn instructions to prevent information leakage. For negative samples (50%), we select v^- from similar users' history $\mathcal{V}_n$ while ensuring no historical interaction exists between u_t and v^- in the training data. This balanced sampling approach maintains unbiased training objectives while preserving crucial collaborative signals for effective profile generation.

$$\begin{aligned}\mathcal{Q}_{sec.} &= \begin{cases} Prompt(u_t, v^+), & pos.\ samp.\\ Prompt(u_t, v^-), & neg.\ samp.\end{cases}\\ \mathcal{R}_{sec.} &= \begin{cases} \text{Yes}, & pos.\ samp.\\ \text{No}, & neg.\ samp.\end{cases}\end{aligned} \tag{4}$$

Tuning Strategy. For multi-turn dialogue instruction-tuning, our objective is to utilize LLM-generated responses $\mathcal{R}$ for weight updates while excluding queries $\mathcal{Q}$. Traditional single-turn dialogue tuning, when applied to our paradigm, considers $\mathcal{Q}fir.$, $\mathcal{R}_{fir.}$, and $\mathcal{Q}_{sec.}$ as inputs, with only $\mathcal{R}_{sec.}$ being predicted. This approach limits weight updates to losses from $\mathcal{R}_{sec.}$ alone, failing to leverage the training data in multi-turn dialogues. Notably, $\mathcal{R}_{fir.}$ contains rich textual information as multiple user profiles, which guides the generation of $\mathcal{R}_{sec.}$ in subsequent turns. In contrast, $\mathcal{R}_{sec.}$ consists of simpler binary responses (*e.g.*, yes or no). Disregarding the valuable information in $\mathcal{R}_{fir.}$ and relying on $\mathcal{R}_{sec.}$ for fine-tuning would significantly limit the model's learning capacity.

To address these limitations, we introduce a two-turn dialogue tuning approach. Specifically, we concatenate dialogues from both turns and apply masking techniques to differentiate between $\mathcal{Q}$ and $\mathcal{R}$ components. During training, the model computes losses exclusively from $\mathcal{R}$-marked segments, enabling both $\mathcal{R}_{fir.}$ and $\mathcal{R}_{sec.}$ to contribute to parameter optimization. This unified training strategy maximizes information utilization while effectively capturing collaborative relationships.

Inference Prompt. Following instruction-tuning, we design an inference prompt that integrates target user interactions $\mathcal{V}_t$ with neighbor histories $\mathcal{V}_n$. This unified prompt guides LLMs to generate collaborative-aware user profiles by leveraging inter-user preference patterns.

$$\mathcal{Q}_{inf.} = Prompt(u_t, \{u_n\}, \mathcal{V}_t, \{\mathcal{V}_n\}). \tag{5}$$

4.3 Refining Profile Generation through Reinforcement Learning

We introduce a reinforcement learning framework aimed at improving the accuracy and personalization of user profiles generated by LLMs. Our approach tackles two key challenges: **i) Prompt-Training Discrepancy**—the inherent mismatch between the inference phase (single-profile generation) and the fine-tuning phase (multi-profile generation), which can lead to inconsistencies in profile generation, and **ii) Personalization-Collaboration Trade-off**—while leveraging collaborative information enhances overall performance, it risks diluting individual user characteristics, analogous to the over-smoothing phenomenon observed in our collaborative instruction-tuning paradigm.

To address these challenges, we develop a reinforcement learning-based fine-tuning paradigm inspired by Reinforcement Learning from Human Feedback (RLHF) (Stiennon et al., 2020). Our approach includes two main components: a reward model that assesses the quality of LLM-generated profiles, and an optimization procedure utilizing Proximal Policy Optimization (PPO) (Schulman et al., 2017) to refine the LLM based on these reward signals. This iterative process gradually improves the LLM's capacity to generate more accurate and personalized user profiles.

Reward model. We design a reward model to evaluate the quality of LLM-generated outputs from a human perspective. Specifically, given an input pair $[\mathcal{Q}, \mathcal{R}]$, the model produces a scalar value indicating the response quality. The reward model is optimized using the loss function $\mathcal{L}_{rm}$:

$$\begin{aligned}\mathcal{L}_{rm} = -\sum_{i=0}^{\mathcal{I}} \mathbb{E}_{(\mathcal{Q}_i, \mathcal{R}_i^+, \mathcal{R}_i^-)\sim D} \\ [\log(\sigma(r_\theta(\mathcal{Q}_i, \mathcal{R}_i^+) - r_\theta(\mathcal{Q}_i, \mathcal{R}_i^-)))],\end{aligned} \tag{6}$$

where $r_\theta(\cdot)$ denotes the reward model, $\sigma(\cdot)$ represents the sigmoid function, and $\mathcal{R}_i^+/\mathcal{R}_i^-$ indicate true/false responses respectively. For our profiling task, we maintain a consistent query $\mathcal{Q}_{inf.}$ while carefully constructing both positive responses $\mathcal{R}^+$ and negative responses $\mathcal{R}^-$. We obtain $\mathcal{R}^+$ through ChatGPT and create $\mathcal{R}^-$ through two strategies:

- **Diverse Negative Sampling:** Multiple prompt templates generate diverse low-quality responses, allowing the reward model to identify suboptimal outputs after instruction-tuning.

- **Profile Substitution:** Strategic replacement of profiles with those of similar users allows the model to identify subtle differences and prioritize personalized characteristics effectively.

Proximal Policy Optimization. In our reinforcement learning framework, we treat the LLM $\mathcal{M}$ as the policy to be optimized, with the reward model approximating the true reward function. The core optimization objective is formulated as:

$$\underset{\mathcal{M}}{\text{argmax}}\, \mathbb{E}_{x_i\sim\mathcal{D}, y_i\sim\mathcal{M}}[R(y_i|x_i)]. \tag{7}$$

To optimize $\mathcal{M}$ iteratively, we sample queries $\mathcal{Q}i$ from the dataset $\mathcal{D}$ and generate corresponding responses $\mathcal{R}i$ using $\mathcal{M}$. We employ the Proximal Policy Optimization (PPO) algorithm and its associated loss function for optimization. Following (Schulman et al., 2017), we enhance the reward function with a KL divergence penalty term between the original LLM $\mathcal{M}_0$ and the optimized LLM $\mathcal{M}_\theta$. This constraint effectively mitigates reward hacking—a phenomenon where models achieve high reward scores but poor human evaluation results. The final reward function $R(\cdot)$ for a given query-response pair $(\mathcal{Q}_i, \mathcal{R}_i)$ is defined as:

$$\begin{aligned}R(\mathcal{R}_i|\mathcal{Q}_i) = \hat{r}(\mathcal{R}_i|\mathcal{Q}_i) - \\ \beta D_{KL}(\mathcal{M}_\theta(\mathcal{Q}_i)||\mathcal{M}_0(\mathcal{Q}_i))\end{aligned} \tag{8}$$

The comprehensive details of our instruction design across all fine-tuning stages, together with our methodology for constructing positive and negative training samples for the reinforcement learning-based reward model, are reported in Appendix A.9.

5 Evaluation

5.1 Experimental Setup

(i) Datasets. To evaluate the effectiveness of the proposed RecLM, we conduct extensive experiments using two public datasets: **MIND**[1] (Wu et al., 2020) and **Netflix**[2], along with a large-scale dataset derived from the real-world industrial data (referred to as **Industrial** for anonymity). **(ii) Evaluation Settings.** We follow previous recommendation works (Jiang et al., 2024) to set evaluation settings. We assess the accuracy of the top-*K* recommendation results using two widely adopted metrics: *Recall@K (R@K)* and *NDCG@K (N@K)*, with *K* set to 20 by default. To reduce bias, we employ an all-rank evaluation strategy, where positive items in the test dataset are ranked alongside all non-interacted items for each user. The final metric is reported as the average result across all users in the test dataset. **(iii) Baseline Methods.** We evaluate the effectiveness of RecLM by integrating it with

[1]https://msnews.github.io

[2]https://www.kaggle.com/datasets/netflix-inc/netflix-prize-data

Table 1: Performance comparison on MIND, Netflix and Industrial data in terms of *Recall* and *NDCG*. The superscript * indicates the improvement is statistically significant where the p-value < 0.05.

Dataset		MIND				Netflix				Industrial			
Backbone	Variants	R@20	R@40	N@20	N@40	R@20	R@40	N@20	N@40	R@20	R@40	N@20	N@40
Full-Shot Setting													
BiasMF	Base	0.0683	0.1039	**0.0311**	0.0399	0.0449	0.0790	0.1451	0.1375	0.0078	0.0143	0.0046	0.0066
	Augment.	**0.0719***	**0.1353***	0.0272	**0.0411***	**0.0531***	**0.0868***	**0.1761***	**0.1630***	**0.0121***	**0.0198***	**0.0074***	**0.0097***
	Improve.	5.27% ↑	30.22% ↑	12.54% ↓	3.01% ↑	18.26% ↑	9.87% ↑	21.36% ↑	18.55% ↑	55.13% ↑	38.46% ↑	60.87% ↑	46.97% ↑
NCF	Base	0.0713	0.0985	**0.0325**	**0.0445**	0.0581	0.0936	0.1848	0.1721	0.0102	0.0076	0.0188	0.0091
	Augment.	**0.0760***	**0.1233***	0.0288	0.0414	**0.0591***	**0.0968***	**0.1903***	**0.1785***	**0.0133***	**0.0087***	**0.0206***	**0.0108***
	Improve.	6.59% ↑	25.18% ↑	11.38% ↓	6.97% ↓	1.72% ↑	3.42% ↑	2.98% ↑	3.72% ↑	30.39% ↑	14.47% ↑	9.57% ↑	18.68% ↑
LightGCN	Base	0.0389	0.0702	0.0150	0.0219	0.0467	0.0815	0.1488	0.1424	0.0096	0.0162	0.0059	0.0076
	Augment.	**0.0788***	**0.0983***	**0.0337***	**0.0384***	**0.0652***	**0.1026***	**0.1703***	**0.1606***	**0.0143***	**0.0225***	**0.0087***	**0.0107***
	Improve.	102.57% ↑	40.03% ↑	124.67% ↑	75.34% ↑	39.61% ↑	25.89% ↑	14.45% ↑	12.78% ↑	48.96% ↑	38.89% ↑	47.46% ↑	40.79% ↑
SGL	Base	0.0345	0.0708	0.0127	0.0210	0.0277	0.0416	0.0855	0.0762	0.0078	0.0138	0.0050	0.0068
	Augment.	**0.0732***	**0.0967***	**0.0367***	**0.0421***	**0.0788***	**0.1204***	**0.1958***	**0.1831***	**0.0133***	**0.0221***	**0.0080***	**0.0106***
	Improve.	112.17% ↑	36.58% ↑	188.98% ↑	100.48% ↑	184.48% ↑	189.42% ↑	129.01% ↑	140.29% ↑	70.51% ↑	60.14% ↑	60% ↑	55.88% ↑
SimGCL	Base	0.0421	0.0636	0.0155	0.0212	0.0231	0.0441	0.0810	0.0825	0.0042	0.0078	0.0026	0.0037
	Augment.	**0.0576***	**0.0908***	**0.0232***	**0.0329***	**0.0567***	**0.0908***	**0.1782***	**0.1673***	**0.0128***	**0.0205***	**0.0080***	**0.0099***
	Improve.	36.82% ↑	42.77% ↑	49.68% ↑	55.19% ↑	145.45% ↑	105.90% ↑	120.00% ↑	102.79% ↑	204.76% ↑	162.82% ↑	207.69% ↑	167.57% ↑
Zero-Shot Setting													
BiasMF	Base	0.0096	0.0165	0.0031	0.0041	0.0311	0.0769	0.0167	0.0292	0.0038	0.0068	0.0020	0.0029
	Augment.	**0.0246***	**0.0373***	**0.0107***	**0.0135***	**0.1381***	**0.1490***	**0.0828***	**0.0584***	**0.0056***	**0.0103***	**0.0026***	**0.0040***
	Improve.	156.25% ↑	126.06% ↑	245.16% ↑	229.27% ↑	344.05% ↑	93.76% ↑	395.81% ↑	100.00% ↑	47.37% ↑	51.47% ↑	30.00% ↑	37.93% ↑
NCF	Base	0.0301	0.0383	0.0080	0.0097	0.0480	0.1158	0.0196	0.0384	0.0044	0.0022	0.0056	0.0026
	Augment.	**0.0424***	**0.0469***	**0.0112***	**0.0122***	**0.1700***	**0.1774***	**0.0984***	**0.0974***	**0.0051***	**0.0031***	**0.0088***	**0.0041***
	Improve.	40.86% ↑	22.45% ↑	40.00% ↑	25.77% ↑	254.17% ↑	53.20% ↑	402.04% ↑	153.65% ↑	15.91% ↑	40.91% ↑	57.14% ↑	57.69% ↑
LightGCN	Base	0.0138	0.0292	0.0046	0.0078	0.0974	0.1256	0.0446	0.0415	0.0092	0.0160	0.0051	0.0070
	Augment.	**0.0196***	**0.0389***	**0.0064***	**0.0086***	**0.1371***	**0.1453***	**0.0697***	**0.0459***	**0.0133***	**0.0188***	**0.0090***	**0.0106***
	Improve.	42.03% ↑	33.22% ↑	39.13% ↑	10.26% ↑	40.76% ↑	15.68% ↑	56.28% ↑	10.60% ↑	44.57% ↑	17.50% ↑	76.47% ↑	51.43% ↑
SGL	Base	0.0162	0.0264	0.0062	0.0074	0.0385	0.1441	0.0274	0.0579	0.0065	0.0114	0.0036	0.0050
	Augment.	**0.0254***	**0.0450***	**0.0089***	**0.0107***	**0.1126***	**0.1756***	**0.0384***	**0.1066***	**0.0111***	**0.0176***	**0.0066***	**0.0084***
	Improve.	56.79% ↑	70.45% ↑	43.55% ↑	44.59% ↑	92.47% ↑	21.86% ↑	40.15% ↑	84.11% ↑	70.77% ↑	54.39% ↑	83.33% ↑	68.00% ↑
SimGCL	Base	0.0164	0.0300	0.0055	0.0084	0.0793	0.1259	0.0336	0.0460	0.0078	**0.0140**	0.0042	0.0059
	Augment.	**0.0312***	**0.0388***	**0.0098***	**0.0115***	**0.1508***	**0.1895***	**0.1550***	**0.1647***	**0.0084***	0.0137	**0.0044***	0.0059
	Improve.	90.24% ↑	29.33% ↑	78.18% ↑	36.90% ↑	90.16% ↑	50.52% ↑	361.31% ↑	258.04% ↑	7.69% ↑	2.14% ↓	4.76% ↑	——

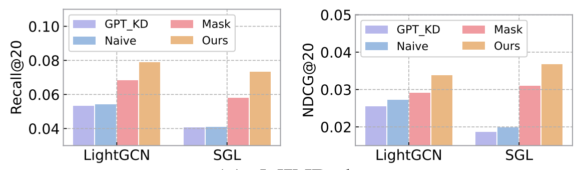
SOTA recommender systems, allowing us to assess performance improvements in a model-agnostic manner compared to baseline models. The selected CF recommenders include non-graph methods such as BiasMF (Koren et al., 2009) and NCF (He et al., 2017), the GNN-enhanced method LightGCN (He et al., 2020), and graph contrastive learning approaches SGL (Wu et al., 2021) and SimGCL (Yu et al., 2022). Details regarding the baselines and datasets are provided in Appendices A.1 and A.2. The experimental results for RecLM presented in the main text are based on llama2-7b-chat as the base LLM. Key results using llama3.1-8b-instruct as the base LLM are provided in Appendix A.7.

5.2 Performance Comparison

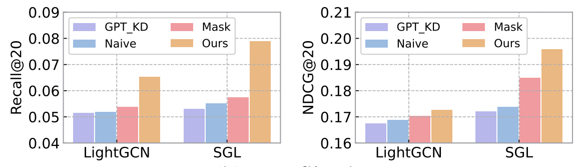
To demonstrate the effectiveness of our RecLM in enhancing performance, particularly in cold-start scenarios, we apply it to five common collaborative filtering methods. The "full-shot" setting corresponds to the complete dataset, while the "zero-shot" setting refers to the pure cold-start condition. The ***Base*** variant applies the cold-start recommendation paradigm to the baseline recommenders without any profiling enhancement via LLMs, whereas the ***Augment*** variant integrates RecLM into the base recommenders. Detailed settings and implementation information are provided in Appendices A.3 and A.5. The evaluation results in Tab. 1 reveal several interesting observations.

(i) Performance Improvement in Integrated Recommenders. We consistently find that integrating RecLM with backbone recommenders leads to enhanced performance compared to the base variant, which relies on raw external item features and ID-based user embeddings in both **supervised** and **zero-shot** settings. This provides strong evidence for the effectiveness of RecLM. We attribute these improvements to two key factors: *First*, for supervised recommendation scenarios, RecLM leverages instruction-tuned LLMs to generate accurate user/item profiles as auxiliary information, effectively enhancing the semantic representation of user preferences. *Second*, our tuning paradigm guides the LLMs in capturing user collaborative relationships, allowing for the generation of high-quality, personalized profiles that demonstrate strong generalization in zero-shot scenarios.

(ii) Outstanding Performance in Cold-Start Scenarios. This improvement arises from our innovative modifications to the ID-embedding paradigm employed in current recommenders. By incorporating external features specifically designed to address the challenges of interaction data scarcity, we have significantly enhanced the effectiveness of these systems. Remarkably, we observe substantial



(a): MIND data



(b): Netflix data

Figure 2: Ablation study on the LLM tuning techniques.

performance improvements even in the relatively sparser MIND and Industrial datasets, where data limitations traditionally pose significant hurdles. By leveraging our RecLM for user and item profiling, we significantly enhance the generalization capabilities of existing recommenders.

(iii) Practicality and Scalability for Real-World Deployment. The results from Industrial dataset demonstrate that RecLM consistently enhances the performance of recommenders in large-scale, highly sparse real-world scenarios. Furthermore, our profile generation methods can be efficiently executed as an offline profiling system to support online applications, making them highly practical for real-world recommendations. To facilitate online recommendation systems, user/item profiles can be updated at regular intervals, such as daily or weekly. The performance improvements observed across various backbone models indicate that RecLM can easily adapt to a range of business models, significantly enhancing their overall effectiveness.

5.3 Ablation Study

We conducted extensive experiments to validate the effectiveness of our proposed instruction tuning techniques by customizing three variants of RecLM: *GPT_KD*, *Naive*, and *Mask*. Detailed descriptions of these variants can be found in Appendix A.4. The results of our experiments are illustrated in Fig. 2 with following conclusions:

(i) Advantage of Collaborative Instruction Tuning. The results in Fig. 2 show that using instruction tuning to capture collaborative relationships among users and items, along with the masking tuning strategy (*i.e.,* the *Mask* variant), significantly enhances performance compared to the *GPT_KD* variant. This improvement suggests that our tuning solution generates more precise, high-quality profiles by leveraging collaborative information effectively. In contrast, profiling based solely on user interaction history has limitations, as it lacks the guidance from collaborative insights. Consequently, this approach often results in less accurate profiles that may include noisy interaction records.

(ii) Effectiveness of the Masking-Based Tuning Strategy. Although the *Naive* variant also employs a two-round dialogue-based instruction tuning technique similar to the *Mask* variant, its improvement over the *GPT_KD* variant is limited. This underscores the advantages of the masking-based tuning strategy, which effectively utilizes responses from the two-round dialogue to update the weights of the LLM and guide its learning of collaborative relationships between users.

(iii) Benefits of Reinforcement Learning-Based Profile Generation Refinement. The results indicate that the *Mask* variant performs significantly worse than RecLM. This finding suggests that the proposed reinforcement learning (RL)-based profile generation refinement technique effectively addresses the noise issues and over-smoothing problems associated with the collaborative instruction-tuning paradigm. As a result, it enables the LLM to generate more accurate and personalized profiles. To intuitively explore the contribution of reinforcement learning to the personalization of generated profiles, we further conducted a case study on the MIND dataset. Details are shown in Appendix A.6.

Table 2: Performance *w.r.t.* various aug. variants.

Dataset		MIND		Netflix	
Backbone	Variants	R@20	N@20	R@20	N@20
LightGCN	Base	0.0389	0.0150	0.0467	0.1488
	w/o User Aug.	0.0302	0.0123	0.0384	0.1213
	w/o Item Aug.	0.0719	0.0287	0.0505	0.1621
	RecLM	**0.0788**	**0.0337**	**0.0652**	**0.1703**
SGL	Base	0.0345	0.0127	0.0277	0.0855
	w/o User Aug.	0.0253	0.0093	0.0173	0.0578
	w/o Item Aug.	0.0719	0.0289	0.0502	0.1546
	RecLM	**0.0732**	**0.0367**	**0.0788**	**0.1958**

5.4 Effectiveness of LLM-enhanced Profiling

To evaluate the impact of our LLM-powered profiling system on user and item feature enhancements, we created two RecLM variants: one without user feature enhancement (*i.e.*, w/o User Aug.) and another without item feature enhancement (*i.e.*, w/o Item Aug.). Experiment results on the MIND and Netflix datasets using LightGCN and SGL as backbone models in the full-shot setting are summarized in Table 2, leading to two major insights:

(i) User-Side Feature Enhancement. Removing user-side enhancements significantly reduces performance across datasets and models, highlighting

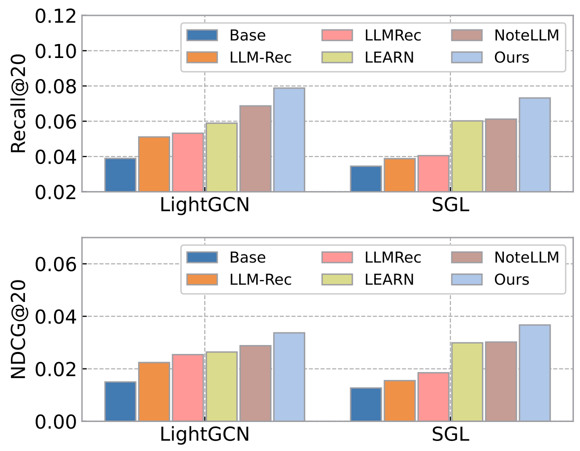


Figure 3: Comparison with LLM-based Recommenders.

the importance of user profiling. Using only ID embeddings cannot effectively model user preferences, while our method excels by extracting text features and integrating graph-text information.

(ii) Item-Side Feature Enhancement. Excluding item-side enhancements also degrades performance. Notably, retaining item enhancements without user enhancements can perform worse than the *Base* variant, due to the complex interaction between raw and enhanced item features. Sole reliance on ID embeddings for users fails to capture their preferences effectively.

5.5 Comparison with LLM-based Methods

We perform a comprehensive benchmarking of our RecLM against existing SOTA LLM-based recommendation systems using the MIND dataset to demonstrate the superiority of our proposed instruction-tuning technique. This comparison includes LLM-Rec (Lyu et al., 2023), LLMRec (Wei et al., 2024), NoteLLM (Zhang et al., 2024), and LEARN (Jia et al., 2024). Specifically, for NoteLLM and LEARN, our evaluation focused on integrating their generated item embeddings with conventional CF recommenders.

The experimental results are presented in Fig. 3. Notably, both LLM-Rec and LLMRec yield suboptimal outcomes, primarily due to their methodology of generating user and item profiles through direct API calls to the LLM, without incorporating task-specific fine-tuning for profile generation. In contrast, NoteLLM and LEARN demonstrate the capability to produce high-quality item embeddings, effectively addressing the cold-start challenge in recommendation systems. However, they have a significant limitation in their ability to adequately capture and leverage the collaborative relationships among users. Therefore, RecLM exhibits substantial performance advantages, leading to significant improvements in the performance of base models.

Table 3: Training efficiency of RecLM.

Dataset	Recommender	Base	RecLM	Cost
MIND	BiasMF	0.72s	0.85s	+18.06%
	NCF	0.76s	0.85s	+11.84%
	LightGCN	0.79s	0.86s	+8.86%
	SGL	1.93s	2.01s	+4.15%
	SimGCL	2.63s	2.69s	+2.28%
Netflix	BiasMF	14.38s	16.42s	+14.19%
	NCF	15.02s	17.17s	+14.31%
	LightGCN	20.47s	20.95s	+2.34%
	SGL	64.98s	65.08s	+0.15%
	SimGCL	44.02s	44.61s	+1.34%
Industrial	BiasMF	7.07s	8.85s	+25.18%
	NCF	7.58s	8.45s	+11.48%
	LightGCN	9.33s	10.25s	+9.86%
	SGL	32.34s	32.87s	+1.64%
	SimGCL	85.41s	86.52s	+1.30%

5.6 Training Efficiency Analysis of RecLM

To assess the efficiency of our RecLM approach, we perform both a theoretical complexity analysis and an empirical running time test. **Theoretical Analysis**: The time complexity of the MLP for transferring textual features $\mathbf{f} \in \mathbb{R}^{d_t}$ into the model's latent space $\mathbb{R}^d$ is $\mathcal{O}(N \times (d_t \times d + d \times d))$, where N is the number of nodes, d_t and d are the dimensions of the text features and latent space, respectively. **Empirical Evaluation**: The per-epoch training time is shown in Tab. 3, conducted on a server with NVIDIA A100 GPUs. Results show that for larger models (*e.g.*, GNN-based methods), RecLM costs less than 10% extra time, and in dense datasets like Netflix, this can be reduced to under 5%. For smaller recommenders, the maximum additional time is about 25%. Given the significant improvements in recommendation performance, the incurred costs are considered acceptable.

6 Conclusion

This work introduces a novel instruction-tuning paradigm that integrates large language models with collaborative filtering, enhancing their ability to capture complex user-item interactions and preferences. This model-agnostic approach easily fits into existing recommender systems, improving generalization in data-scarce scenarios where traditional methods falter. Key innovations include combining external features with collaborative patterns and a reinforcement learning-based framework for personalized feature enhancement, tackling cold-start profiling and data noise challenges. Evaluations demonstrate the significant benefits and compatibility of our approach with state-of-the-art recommender systems.

7 Limitations

In real-world scenarios, items commonly have abundant modal information, including text, images, audio, and more. However, this work primarily focuses on exploring the collaborative feature enhancement paradigm based on textual features, and does not fully exploit the potential of multimodal information. While the proposed method can be extended to other modalities using distinct modal encoders, it is important to note that other modalities may introduce novel challenges and opportunities for feature enhancement. Thus, the exploration of these modalities represents a promising future direction for further investigation.

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 *RecSys*, pages 1007–1014.

Junyi Chen, Lu Chi, Bingyue Peng, and Zehuan Yuan. 2024. Hllm: Enhancing sequential recommendations via hierarchical large language models for item and user modeling. *arXiv preprint arXiv:2409.12740*.

Lei Chen, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. 2020. Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. In *AAAI*, volume 34, pages 27–34.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. 2023. Recommender systems in the era of large language models (llms). *arXiv preprint arXiv:2307.02046*.

Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-rec: Towards interactive and explainable llms-augmented recommender system. *arXiv preprint arXiv:2303.14524*.

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 *RecSys*, pages 299–315.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*, pages 639–648.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*, pages 173–182.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *RecSys*, pages 135–142.

Jian Jia, Yipei Wang, Yan Li, Honggang Chen, Xuehan Bai, Zhaocheng Liu, Jian Liang, Quan Chen, Han Li, Peng Jiang, et al. 2024. Knowledge adaptation from large language model to recommendation for practical industrial application. *arXiv preprint arXiv:2405.03988*.

Yangqin Jiang, Chao Huang, and Lianghao Huang. 2023. Adaptive graph contrastive learning for recommendation. In *SIGKDD*, pages 4252–4261.

Yangqin Jiang, Lianghao Xia, Wei Wei, Da Luo, Kangyi Lin, and Chao Huang. 2024. Diffmm: Multi-modal diffusion model for recommendation. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 7591–7599.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Yehuda Koren, Steffen Rendle, and Robert Bell. 2021. Advances in collaborative filtering. *Recommender systems handbook*, pages 91–142.

Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Xiangyang Li, Chenxu Zhu, Huifeng Guo, Yong Yu, Ruiming Tang, et al. 2023. How can recommender systems benefit from large language models: A survey. *arXiv preprint arXiv:2306.05817*.

Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *WWW*, pages 2320–2329.

Peng Liu, Lemei Zhang, and Jon Atle Gulla. 2023. Pre-train, prompt, and recommendation: A comprehensive survey of language modeling paradigm adaptations in recommender systems. *Transactions of the Association for Computational Linguistics*, 11:1553–1571.

Sichun Luo, Bowei He, Haohan Zhao, Wei Shao, Yanlin Qi, Yinya Huang, Aojun Zhou, Yuxuan Yao, Zongpeng Li, Yuanzhang Xiao, et al. 2024. Recranker: Instruction tuning large language model as ranker for top-k recommendation. *ACM TOIS*.

Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Christopher Leung, Jiajie Tang, and Jiebo Luo. 2023. Llm-rec: Personalized recommendation via prompting large language models. *arXiv preprint arXiv:2307.15780*.

Xubin Ren, Wei Wei, Lianghao Xia, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2023. Representation learning with large language models for recommendation. *arXiv preprint arXiv:2310.15950*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *NIPS*, 33:3008–3021.

Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Jianling Wang, Raphael Louca, Diane Hu, Caitlin Cellier, James Caverlee, and Liangjie Hong. 2020. Time to shop for valentine's day: Shopping occasions and sequential recommendation in e-commerce. In *WSDM*, pages 645–653.

Xiang Wang, Xiangnan He, Meng Wang, et al. 2019. Neural graph collaborative filtering. In *SIGIR*.

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 *WSDM*, pages 806–815.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, et al. 2020. Mind: A large-scale dataset for news recommendation. In *ACL*, pages 3597–3606.

Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *SIGIR*, pages 726–735.

Liang Wu, Diane Hu, Liangjie Hong, and Huan Liu. 2018. Turning clicks into purchases: Revenue optimization for product search in e-commerce. In *SIGIR*, pages 365–374.

Tiansheng Yao, Xinyang Yi, Derek Zhiyuan Cheng, Felix Yu, Ting Chen, Aditya Menon, Lichan Hong, Ed H Chi, Steve Tjoa, Jieqi Kang, et al. 2021. Self-supervised learning for large-scale item recommendations. In *CIKM*, pages 4321–4330.

Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Jundong Li, and Zi Huang. 2023. Self-supervised learning for recommender systems: A survey. *TKDE*.

Junliang Yu, Hongzhi Yin, Xin Xia, et al. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *SIGIR'22*.

Zheng Yuan, Fajie Yuan, Yu Song, Youhua Li, Junchen Fu, Fei Yang, Yunzhu Pan, and Yongxin Ni. 2023. Where to go next for recommender systems? id-vs. modality-based recommender models revisited. In *SIGIR*, pages 2639–2649.

Ruohan Zhan, Changhua Pei, Qiang Su, Jianfeng Wen, Xueliang Wang, Guanyu Mu, Dong Zheng, Peng Jiang, and Kun Gai. 2022. Deconfounding duration bias in watch-time prediction for video recommendation. In *SIGKDD*, pages 4472–4481.

Chao Zhang, Shiwei Wu, Haoxin Zhang, Tong Xu, Yan Gao, Yao Hu, and Enhong Chen. 2024. Notellm: A retrievable large language model for note recommendation. In *WWW*, pages 170–179.

Fanjin Zhang, Jie Tang, Xueyi Liu, Zhenyu Hou, Yuxiao Dong, Jing Zhang, Xiao Liu, Ruobing Xie, Kai Zhuang, Xu Zhang, et al. 2021. Understanding wechat user preferences and "wow" diffusion. *TKDE*, 34(12):6033–6046.

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*.

A Appendix / supplemental material

A.1 Details of Dataset

Table 4 provides a summary of the statistical information for the three datasets. The following sections outline the specific details for each dataset:

- **MIND**: This large-scale dataset is designed for news recommendation research. We selected data from two consecutive days, assigning one day as the training set and the other as the test set. The raw text includes the news category, title, and abstract.

- **Netflix**: It is selected from a renowned video streaming platform, and we get the implicit feedback data from the Netflix Prize Data on Kaggle. We curated two consecutive years' worth of data based on time, utilizing one year as the training set and the other as the test set. The raw text information for the items was derived from the movie titles themselves.

- **Industrial**: It is a large-scale real dataset, which is collected from a prominent online content platform (name omitted for anonymity), serving millions of users. It comprises news articles. We sampled data from two consecutive dates, assigning them as the training set and test set, respectively. The raw text information for each item is represented by its title.

Table 4: Statistics of the experimental datasets.

Statistics	**MIND**	**Netflix**	**Industrial**
# User	57128	16835	117433
# Overlap. Item	1020	6232	72417
# Snapshot	daily	yearly	daily
Training Set			
# Item	2386	6532	152069
# Interactions	89734	1655395	858087
# Sparsity	99.934%	98.495%	99.995%
Test Set			
# Item	2461	8413	158155
# Interactions	87974	1307051	876415
# Sparsity	99.937%	99.077%	99.995%

A.2 Details of Selected Base Models

This section gives a brief introduction of the selected base models in this work.

- **BiasMF** (Koren et al., 2009): It is a matrix factorization method that aims to enhance user-specific preferences for recommendation by incorporating bias vectors for users and items.

- **NCF** (He et al., 2017): It is a neural network-based method that replaces the dot-product operation in conventional matrix factorization with multi-layer neural networks. This allows the model to capture complex user-item interactions and provide recommendations. For our comparison, we utilize the NeuMF variant of NCF.

- **LightGCN** (He et al., 2020): This model leverages the power of neighborhood information in the user-item interaction graph by using a layer-wise propagation scheme that involves only linear transformations and element-wise additions.

- **SGL** (Wu et al., 2021): The model enhances LightGCN by integrating contrastive learning with self-supervision. It employs data augmentation strategies, including random walks and node/edge dropout, to corrupt graph structures.

- **SimGCL** (Yu et al., 2022): This work introduces a straightforward contrastive learning (CL) method that eliminates graph augmentations. Instead, it adds uniform noise to the embedding space to generate contrastive views.

A.3 Performance Comparison: Setting

In the performance comparison experiments outlined in Sec. 5.2, we considered two distinct testing data settings: the full-shot setting and the zero-shot setting. The full-shot setting entailed using the original test set as the testing data, where certain items in the test set had appeared in the training set previously. Conversely, the zero-shot setting involved exclusively testing items that had not been encountered in the training set. This setting was specifically designed to assess the effectiveness of our proposed RecLM in addressing the item cold-start scenario, where limited or no prior information is available for certain items.

In the conducted experiments, we explored two variants: *Base* and *Augment*. The *Base* variant demonstrates the application of our proposed cold-start recommendation paradigm by utilizing only user-side ID embeddings and item-side raw text embeddings, without incorporating the profiles generated by LLMs. On the other hand, the *Augment* variant involves fully integrating our proposed RecLM into traditional recommenders. The comparison between two variants enables us to assess the effectiveness of our approach in enhancing the performance of recommenders by leveraging LLMs to generate informative profiles.

A.4 Ablation Study: Setting

In the case of *GPT_KD* variant, the approach involves exclusively fine-tuning the open-source LLM by utilizing user profile data generated solely through ChatGPT3.5, as discussed in Sec. 4.2.1. Conversely, for *Naive* variant, the two-turn dialogue-based instruction tuning technique (*i.e.,* Sec. 4.2.2) is applied based on the variant *GPT_KD*, but with the tuning strategy limited to the conventional single-turn dialogue tuning approach. As for the variant *Mask*, a similar two-turn dialogue-based instruction tuning technique is employed based on the variant *GPT_KD*, with the additional application of a masking-based tuning strategy. As for *Ours*, it refers to RecLM, which employs RL-based personalized feature enhancement based on the variant *Mask*.

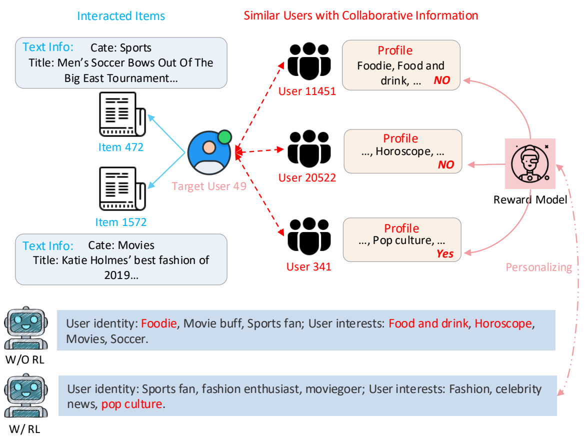
A.5 Implementation Details

A.5.1 Parameter-Efficient Fine-Tuning

To achieve efficient fine-tuning of LLMs while preserving their inherent knowledge reasoning capabilities, we employed the Parameter-Efficient Fine-Tuning (PEFT) method. Specifically, in this study, we chose Low-Rank Adaptation (LoRA) (Hu et al., 2021) as the fine-tuning technique for the open-source LLMs, specifically Llama2-7b-chat (Touvron et al., 2023) and Llama3-8b-instruct (Dubey et al., 2024). This approach allows us to strike a balance between retaining the valuable knowledge of the pre-trained models and adapting them to specific tasks effectively.

A.5.2 Integration of RecLM into Various Base Recommenders

Following the integration of our method into various base recommenders, we meticulously conducted an extensive hyperparameter search, and also explored the optimal approach for incorporating profile features for each recommendation methods, ensuring a fair comparison. Specifically, each base model is implemented with PyTorch, using Adam optimizer and Xavier initializer with default parameters. Training batch size is set as 4096. The dimensionality of embedding vectors is


Interacted Items
Similar Users with Collaborative Information
Text Info: Cate: Sports
Title: Men's Soccer Bows Out Of The Big East Tournament...
Item 472
Item 1572
Text Info: Cate: Movies
Title: Katie Holmes' best fashion of 2019...
Target User 49
User 11451
User 20522
User 341
Profile
Foodie, Food and drink, ... NO
Profile
..., Horoscope, ... NO
Profile
..., Pop culture, ... Yes
Reward Model
Personalizing
W/O RL
W/ RL

User identity: Foodie, Movie buff, Sports fan; User interests: Food and drink, Horoscope, Movies, Soccer.

User identity: Sports fan, fashion enthusiast, moviegoer; User interests: Fashion, celebrity news, pop culture.

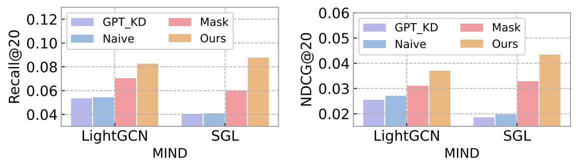
Figure 4: Generated profiles w/ and w/o RL.

set as 32. The learning rate is set as $1e-3$. The coefficient for controling $\mathcal{L}_2$ regularization term is searched in $\{1e-3, 1e-4, 1e-5, 1e-6, 1e-7\}$. For GNN-based models (*e.g.,* LightGCN, SGL, and SimGCL), the number of GCN layers is set as 2. For SSL-based models (*e.g.,* SGL and SimGCL), the temperature coefficient is searched in $\{0.1, 0.5, 1.0\}$.

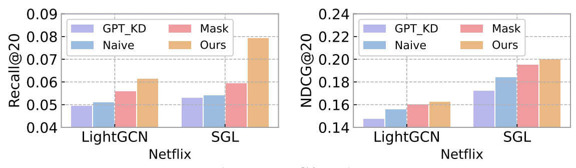
A.6 Case Study

To intuitively explore the contribution of reinforcement learning to the personalization of generated profiles, we conducted a case study using the MIND dataset. In this study, as shown in Fig. 4, the target user for whom the profile is being generated is *User 49*. This user has interacted with two items: *Item 472* and *Item 1572*. Additionally, we identified three similar users who provide collaborative information: *User 11451*, *User 20522*, and *User 341*.

The user profile generated for *User 49* after instruction tuning, but without RL tuning, contains several irrelevant keywords related to the interacted items, such as "Foodie," "Food and drink," and "Horoscope." Notably, these terms also appear in the profiles of *User 11451* and *User 20522*, suggesting that the generated profile is overly influenced by too many collaborative users. In contrast, the profile generated for *User 49* after RL tuning effectively preserves the preferences indicated in the interaction history while incorporating relevant implicit keywords from collaborative users. For example, the term "pop culture" is derived from *User 341*'s profile. This approach provides precise and valuable additional information for modeling *User 49*'s preferences. We attribute this improvement to our proposed RL-based personalized feature en-



(a): MIND data



(b): Netflix data

Figure 5: Ablation study on the LLM tuning techniques.

hancement techniques, which effectively address the noise and over-smoothing issues that can arise during the instruction-tuning process.

A.7 key Experimental Results Obtained Using Llama3.1-8b-instruct as the Base LLM.

A.7.1 Performance Comparison

To validate that our RecLM can achieve outstanding results on different open-source LLMs, we conduct relevant experiments on the latest open-source LLM, llama3.1-8b-instruct. The settings are consistent with the experimental settings in the main text. The relevant experimental results are shown in Table 5.

The experimental results indicate that our RecLM can effectively enhance the performance of traditional collaborative filtering recommendation system models in cold-start scenarios, demonstrating the generalizability of the instruction fine-tuning approach we proposed, which can be applied to various open-source LLMs. Furthermore, when comparing the results based on llama2-7b-chat with those from llama3.1-8b-instruct, we observe improvements across most metrics. This suggests that as the performance of the base LLMs increases, our method can deliver even greater enhancements for recommendation systems.

A.7.2 Ablation Study

We also conduct ablation experiments on the proposed instruction tuning technique based on llama3.1-8b-instruct, and the specific results are shown in Figure 5. The experimental results clearly demonstrate that the key tuning techniques we proposed are effective across various open-source LLMs, reinforcing the generalizability of our method.

A.8 Algorithmic Description

Algorithm 1: Text-enhanced representations for ID-based recommendation framework.

```
Input: User set $\mathcal{U}$, item set $\mathcal{V}$, item text features $F_v$, user interaction
       histories $F_u$, user id embeddings $E_u$, and item id embeddings $E_v$.
Output: Augmented representations $\hat{e}_u^{aug}$, $\hat{e}_v^{aug}$.
for each item $v \in \mathcal{V}$ do
    $f_v = \text{MLP}_{text}(F_v[v])$
end
Generate user profiles $P_u = \text{LLM}_{profile}(F_u)$
Generate item profiles $P_v = \text{LLM}_{profile}(F_v)$
for each user $u \in \mathcal{U}$ do
    $\hat{e}_u^{aug} = \text{MLP}_{fusion}(E_u[u] || \text{MLP}_{proj}(P_u[u]))$
end
for each item $v \in \mathcal{V}$ do
    $\hat{e}_v^{aug} = \text{MLP}_{fusion}(f_v[v] || \text{MLP}_{proj}(P_v[v]))$
end
Initialize recommender **R** with parameters $\theta$
for epoch $= 1$ to $T$ do
    for $(u, v)$ in interaction graph $G$ do
        $\hat{y} = \hat{e}_u^{aug} \cdot \hat{e}_v^{aug}$
        $\mathcal{L} = \text{BPR\_Loss}(\hat{y}, D)$
        Update $\theta$ via Adam
    end
end
Return $\hat{e}_u^{aug}$, $\hat{e}_v^{aug}$
```

A.9 Instruction Designs

In this section, we provide a comprehensive overview of the instructions utilized for fine-tuning at each stage of our process. We will also discuss the methodologies employed to construct both positive and negative training samples for the reinforcement learning reward model.

Instruction designs for ChatGPT knowledge distillation. As shown in Figure 6(a), to facilitate the knowledge distillation process of ChatGPT, we leverage the textual information associated with each user and the items they interact with as inputs for the LLMs. The LLMs then generate user profiles, encompassing the user's identity along with

Table 5: Performance comparison on MIND and Netflix data in terms of *Recall* and *NDCG*. The superscript * indicates the improvement is statistically significant where the p-value < 0.05.

Dataset		MIND				Netflix			
Backbone	Variants	R@20	R@40	N@20	N@40	R@20	R@40	N@20	N@40
Full-Shot Setting									
BiasMF	Base	0.0683	0.1039	0.0311	0.0399	0.0449	0.0790	0.1451	0.1375
	Augment.	**0.0839***	**0.1402***	**0.0331***	**0.0479***	**0.0559***	**0.0871***	**0.1802***	**0.1689***
NCF	Base	0.0713	0.0985	**0.0325**	**0.0445**	0.0581	0.0936	0.1848	0.1721
	Augment.	**0.0789***	**0.1291***	0.0312	0.0442	**0.0656***	**0.1031***	**0.1987***	**0.1823***
LightGCN	Base	0.0389	0.0702	0.0150	0.0219	0.0467	0.0815	0.1488	0.1424
	Augment.	**0.0823***	**0.1021***	**0.0371***	**0.0424***	**0.0613***	**0.0978***	**0.1622***	**0.1496***
SGL	Base	0.0345	0.0708	0.0127	0.0210	0.0277	0.0416	0.0855	0.0762
	Augment.	**0.0877***	**0.1123***	**0.0433***	**0.0539***	**0.0792***	**0.1287***	**0.1999***	**0.1871***
SimGCL	Base	0.0421	0.0636	0.0155	0.0212	0.0231	0.0441	0.0810	0.0825
	Augment.	**0.0617***	**0.0994***	**0.0281***	**0.0343***	**0.0613***	**0.1013***	**0.1872***	**0.1730***
Zero-Shot Setting									
BiasMF	Base	0.0096	0.0165	0.0031	0.0041	0.0311	0.0769	0.0167	0.0292
	Augment.	**0.0310***	**0.0446***	**0.0169***	**0.0192***	**0.1211***	**0.1370***	**0.0797***	**0.0554***
NCF	Base	0.0301	0.0383	0.0080	0.0097	0.0480	0.1158	0.0196	0.0384
	Augment.	**0.0437***	**0.0489***	**0.0136***	**0.0141***	**0.1813***	**0.1897***	**0.1037***	**0.0994***
LightGCN	Base	0.0138	0.0292	0.0046	0.0078	0.0974	0.1256	0.0446	0.0415
	Augment.	**0.0211***	**0.0417***	**0.0071***	**0.0100***	**0.1425***	**0.1601***	**0.0749***	**0.0481***
SGL	Base	0.0162	0.0264	0.0062	0.0074	0.0385	0.1441	0.0274	0.0579
	Augment.	**0.0278***	**0.0481***	**0.0103***	**0.0119***	**0.1210***	**0.1880***	**0.0515***	**0.0950***
SimGCL	Base	0.0164	0.0300	0.0055	0.0084	0.0793	0.1259	0.0336	0.0460
	Augment.	**0.0297***	**0.0354***	**0.0088***	**0.0103***	**0.1654***	**0.2003***	**0.1668***	**0.1694***

their respective interests.

Instruction designs for two-turn dialogue instruction tuning. For the instruction-tuning based on two-round dialogues, meticulous attention has been given to designing corresponding instructions. As illustrated in Figure 6(b), we commence by providing specific system instructions to stimulate the LLMs' comprehension of collaborative filtering methods. Subsequently, in the first round of dialogue, the input instructions encompass the interaction history of several similar users, along with the relevant details of the items involved. To obtain those similar users, we employ a conventional ID-based collaborative filtering recommendation system, followed by similarity calculation based on these embeddings. The expected output from the LLMs should include user profiles for each mentioned user in the input. Moving on to the second round of dialogue, we explicitly prompt the LLMs to determine, based on the acquired user profiles and item information, whether a previously mentioned item is likely to be interacted with by a specific user using collaborative filtering methods. The expected response from the LLMs should be a binary "Yes" or "No" answer.

Instruction designs for user profile generation. Once the instruction-tuning stage is complete, the LLMs are equipped with the capability to generate profiles while considering collaborative relationships. In line with Figure 6(c), we have meticulously designed instructions specifically for user profile generation. Consistent with the instruction-tuning stage, we provide explicit system instructions to stimulate the LLMs' comprehension of collaborative filtering methods. The input instructions encompass the interaction records of multiple similar users (including a target user for whom the LLMs are required to generate a profile) as well as detailed textual information pertaining to the involved items. The expected output from the LLMs is the target user profile.

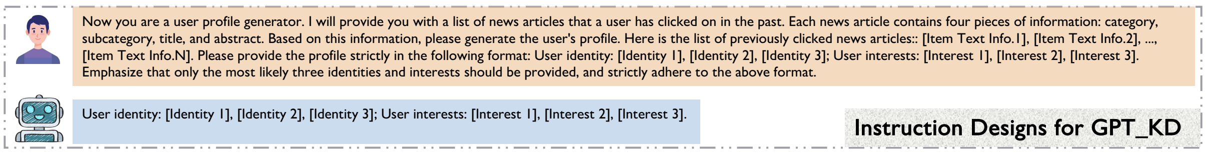
Instruction designs for item profile generation. To ensure semantic alignment between user-side and item-side features, our next objective, after obtaining high-quality user profiles, is to generate item profiles based on the user's profile. Here, the item profile refers to the profile of the target user for that particular item. To accomplish this, we adopt a two-step approach. Firstly, for items that have user interactions, we generate item profiles by leveraging the profiles of the interacting users. This helps establish a connection between the users and the items they engage with. Secondly, using the raw embeddings of the items, we search for similar cold-start items and employ the LLM to infer their profiles based on semantic similarity. As depicted in Figure 6(d), the input instructions consist of a target item and several similar items. We provide the specific textual information of these items, along

Algorithm 2: LLM Instruction Tuning with Collaborative Signals.

```
Input: Pretrained language model $LLM$, user-item interaction graph $G$, and user collaborative neighborhoods $\mathcal{N}_u$.
Output: Fine-tuned LLM model.
# Two-phase Instruction Construction
for each user $u \in G.nodes$ do
    # Phase 1: Collaborative Profile Generation
    $Q^{(1)} = FormatPrompt(u, \mathcal{N}_\sqcap, \mathcal{V}_u, \mathcal{V}_{\mathcal{N}_u})$
    $R^{(1)} = LLM_{generate}(Q^{(1)})$
    # Phase 2: Interaction Prediction
    $v^* = SampleBalanced(\mathcal{V}_u, \mathcal{V}_{\mathcal{N}_u})$
    $Q^{(2)} = FormatPredPrompt(u, v^*)$
    $R^{(2)} = \mathbb{1}[(u, v^*) \in G.edges]$
    # Masked Sequence Training
    $X = [Q^{(1)}; R^{(1)}; Q^{(2)}]$
    Compute $\mathcal{L}_{CE} = CE(LLM(X), Mask(R^{(1:2)}))$
end
# RL-based Refinement
Initialize $R_\psi$ via contrastive learning
for $k = 1$ to $K$ do
    Generate responses $\pi = LLM(Q^{infer})$
    Compute $r = R_\psi(Q^{infer}, \pi) - \beta KL(\pi || \pi_{ref})$
    Update LLM via PPO with $\mathcal{L}^{CLIP}(r)$
end
Return $LLM_{fine-tuned}$
```

with the profiles of the similar items (selected from items that already have profiles). The expected output from the LLMs is the profile of the target item, further enhancing semantic alignment across the recommendation system.

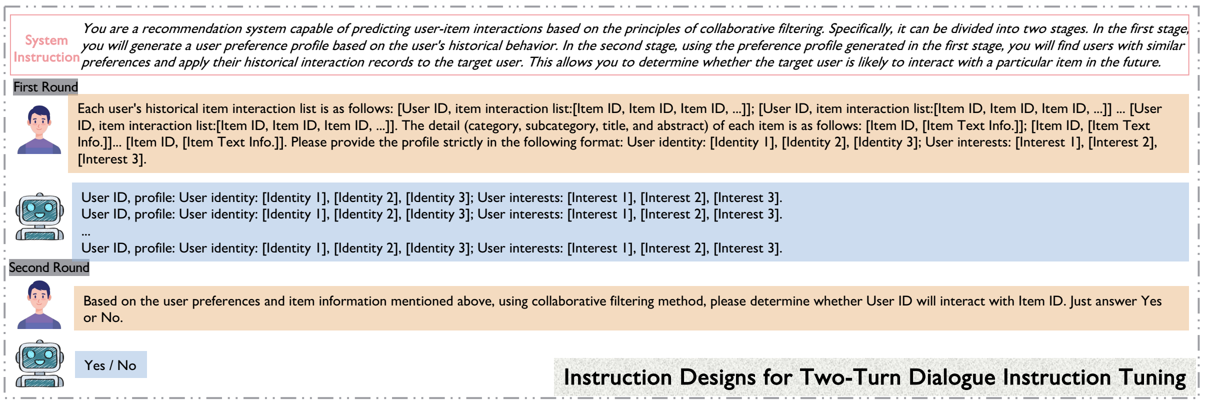
Positive/Negative responses construction for reward model training. In Sec 4.3, we propose personalized feature enhancement based on reinforcement learning as a means to address the noise introduced by instruction-tuning and the potential over-smoothing issue stemming from collaborative feature enhancement. The crux of reinforcement learning lies in training the reward model, and constructing high-quality positive and negative samples plays a pivotal role in this process. As shown in Figure 6(e), for positive samples, we leverage SOTA LLMs (*e.g.*, ChatGPT) with a manual selection approach. For negative samples, they can be categorized into two distinct groups. The first category consists of profiles of similar users, which aim to train the reward model in distinguishing more nuanced profiles and mitigating the over-smoothing issue. The second category encompasses low-quality responses of various types, such as missing or repeated profiles, thereby providing negative examples for training the reward model effectively.


Now you are a user profile generator. I will provide you with a list of news articles that a user has clicked on in the past. Each news article contains four pieces of information: category, subcategory, title, and abstract. Based on this information, please generate the user's profile. Here is the list of previously clicked news articles:: [Item Text Info.1], [Item Text Info.2], ..., [Item Text Info.N]. Please provide the profile strictly in the following format: User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3]. Emphasize that only the most likely three identities and interests should be provided, and strictly adhere to the above format.

User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3].

Instruction Designs for GPT_KD

(a) Instruction designs for ChatGPT knowledge distillation.


System Instruction

You are a recommendation system capable of predicting user-item interactions based on the principles of collaborative filtering. Specifically, it can be divided into two stages. In the first stage, you will generate a user preference profile based on the user's historical behavior. In the second stage, using the preference profile generated in the first stage, you will find users with similar preferences and apply their historical interaction records to the target user. This allows you to determine whether the target user is likely to interact with a particular item in the future.

First Round

Each user's historical item interaction list is as follows: [User ID, item interaction list:[Item ID, Item ID, Item ID, ...]]; [User ID, item interaction list:[Item ID, Item ID, Item ID, ...]] ... [User ID, item interaction list:[Item ID, Item ID, Item ID, ...]]. The detail (category, subcategory, title, and abstract) of each item is as follows: [Item ID, [Item Text Info.]]; [Item ID, [Item Text Info.]]... [Item ID, [Item Text Info.]]. Please provide the profile strictly in the following format: User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3].

User ID, profile: User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3].
User ID, profile: User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3].
...
User ID, profile: User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3].

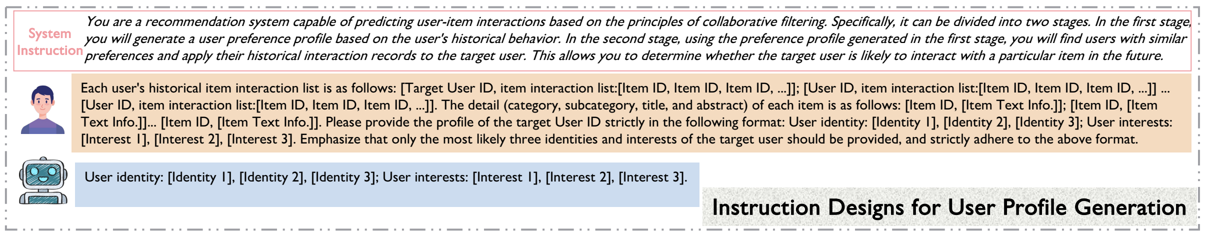
Second Round

Based on the user preferences and item information mentioned above, using collaborative filtering method, please determine whether User ID will interact with Item ID. Just answer Yes or No.

Yes / No

Instruction Designs for Two-Turn Dialogue Instruction Tuning

(b) Instruction designs for two-turn dialogue instruction tuning.


System Instruction

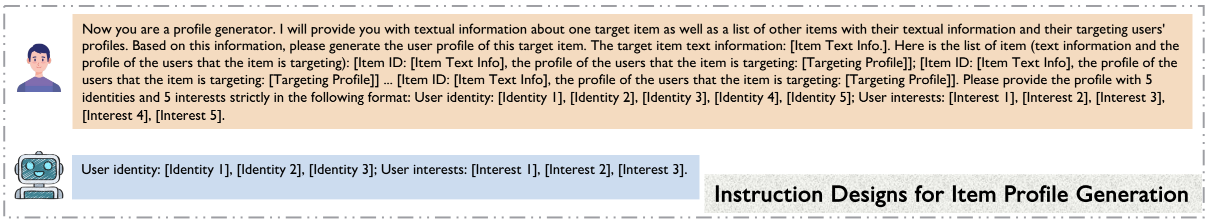
You are a recommendation system capable of predicting user-item interactions based on the principles of collaborative filtering. Specifically, it can be divided into two stages. In the first stage, you will generate a user preference profile based on the user's historical behavior. In the second stage, using the preference profile generated in the first stage, you will find users with similar preferences and apply their historical interaction records to the target user. This allows you to determine whether the target user is likely to interact with a particular item in the future.

Each user's historical item interaction list is as follows: [Target User ID, item interaction list:[Item ID, Item ID, Item ID, ...]]; [User ID, item interaction list:[Item ID, Item ID, Item ID, ...]] ... [User ID, item interaction list:[Item ID, Item ID, Item ID, ...]]. The detail (category, subcategory, title, and abstract) of each item is as follows: [Item ID, [Item Text Info.]]; [Item ID, [Item Text Info.]]... [Item ID, [Item Text Info.]]. Please provide the profile of the target User ID strictly in the following format: User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3]. Emphasize that only the most likely three identities and interests of the target user should be provided, and strictly adhere to the above format.

User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3].

Instruction Designs for User Profile Generation

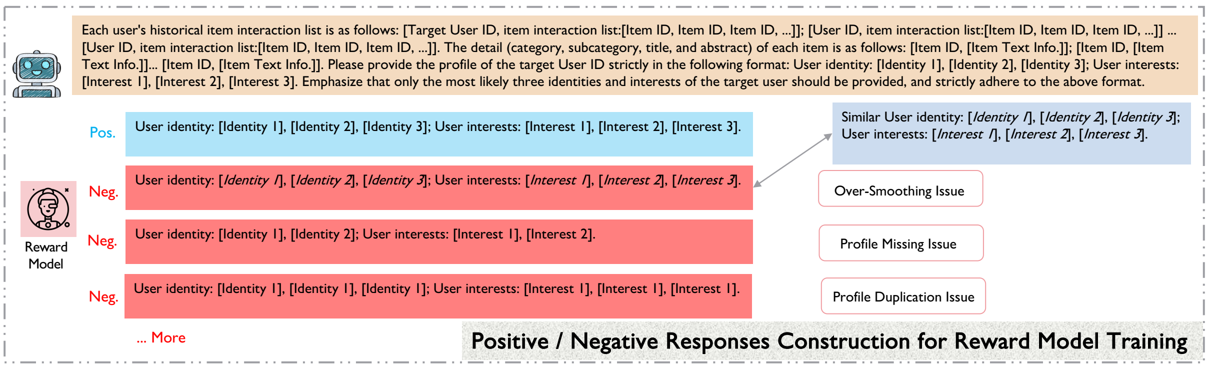
(c) Instruction designs for user profile generation.


Now you are a profile generator. I will provide you with textual information about one target item as well as a list of other items with their textual information and their targeting users' profiles. Based on this information, please generate the user profile of this target item. The target item text information: [Item Text Info.]. Here is the list of item (text information and the profile of the users that the item is targeting): [Item ID: [Item Text Info], the profile of the users that the item is targeting: [Targeting Profile]]; [Item ID: [Item Text Info], the profile of the users that the item is targeting: [Targeting Profile]] ... [Item ID: [Item Text Info], the profile of the users that the item is targeting: [Targeting Profile]]. Please provide the profile with 5 identities and 5 interests strictly in the following format: User identity: [Identity 1], [Identity 2], [Identity 3], [Identity 4], [Identity 5]; User interests: [Interest 1], [Interest 2], [Interest 3], [Interest 4], [Interest 5].

User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3].

Instruction Designs for Item Profile Generation

(d) Instruction designs for item profile generation.


Each user's historical item interaction list is as follows: [Target User ID, item interaction list:[Item ID, Item ID, Item ID, ...]]; [User ID, item interaction list:[Item ID, Item ID, Item ID, ...]] ... [User ID, item interaction list:[Item ID, Item ID, Item ID, ...]]. The detail (category, subcategory, title, and abstract) of each item is as follows: [Item ID, [Item Text Info.]]; [Item ID, [Item Text Info.]]... [Item ID, [Item Text Info.]]. Please provide the profile of the target User ID strictly in the following format: User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3]. Emphasize that only the most likely three identities and interests of the target user should be provided, and strictly adhere to the above format.

Pos. User identity: [Identity 1], [Identity 2], [Identity 3]; User interests: [Interest 1], [Interest 2], [Interest 3].

Neg. User identity: [*Identity 1*], [*Identity 2*], [*Identity 3*]; User interests: [*Interest 1*], [*Interest 2*], [*Interest 3*].

Neg. User identity: [Identity 1], [Identity 2]; User interests: [Interest 1], [Interest 2].

Neg. User identity: [Identity 1], [Identity 1], [Identity 1]; User interests: [Interest 1], [Interest 1], [Interest 1].

... More

Similar User identity: [*Identity 1*], [*Identity 2*], [*Identity 3*]; User interests: [*Interest 1*], [*Interest 2*], [*Interest 3*].

Positive / Negative Responses Construction for Reward Model Training

(e) Positive/Negative responses construction for reward model training.

Figure 6: Instruction designs for RecLM