TLRec: A Transfer Learning Framework to Enhance Large Language Models for Sequential Recommendation Tasks

Jiaye Lin*
Tsinghua University
China
lin-jy22@mails.tsinghua.edu.cn

Shuang Peng†
Zhejiang Lab
China
pengshuang92@gmail.com

Zhong Zhang
Tencent AI Lab
China
zz.ustc@gmail.com

Peilin Zhao
Tencent AI Lab
China
masonzhao@tencent.com

Abstract

Recently, Large Language Models (LLMs) have garnered significant attention in recommendation systems, improving recommendation performance through in-context learning or parameter-efficient fine-tuning. However, cross-domain generalization, i.e., model training in one scenario (source domain) but inference in another (target domain), is underexplored. In this paper, we present TLRec, a transfer learning framework aimed at enhancing LLMs for sequential recommendation tasks. TLRec specifically focuses on text inputs to mitigate the challenge of limited transferability across diverse domains, offering promising advantages over traditional recommendation models that heavily depend on unique identities (IDs) like user IDs and item IDs. Moreover, we leverage the source domain data to further enhance LLMs' performance in the target domain. Initially, we employ powerful closed-source LLMs (e.g., GPT-4) and chain-of-thought techniques to construct instruction tuning data from the third-party scenario (source domain). Subsequently, we apply curriculum learning to fine-tune LLMs for effective knowledge injection and perform recommendations in the target domain. Experimental results demonstrate that TLRec achieves superior performance under the zero-shot and few-shot settings.

CCS Concepts

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

Keywords

Sequential Recommendation, Large Language Models, Transfer Learning, Chain-of-Thought, Instruction Tuning

ACM Reference Format:

Jiaye Lin, Shuang Peng, Zhong Zhang, and Peilin Zhao. 2024. TLRec: A Transfer Learning Framework to Enhance Large Language Models for Sequential Recommendation Tasks. In *18th ACM Conference on Recommender Systems (RecSys '24), October 14–18, 2024, Bari, Italy*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3640457.3691710

*Work done in Tencent AI Lab.
†Corresponding author.

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

ACM ISBN 979-8-4007-0505-2/24/10
https://doi.org/10.1145/3640457.3691710

1 Introduction

Large Language Models (LLMs) have exhibited remarkable capability in simulating human language generation and have found extensive applications across various tasks, such as natural language understanding [7, 8, 36], dialogue systems [4, 17, 27], and code generation [22, 23, 42]. Recent studies highlight the potential of harnessing abundant knowledge resources in LLMs to enhance the generalization of current recommendation systems [31, 32, 37]. By providing tailored instructions, LLMs enable effective adaptation to new tasks, which holds substantial promise.

Conventional recommendation systems employ Transformer-based models [5, 28], incorporating users' historical behavior sequences to capture their preferences for improved performance [21, 24, 43, 44]. While these approaches show promising results, they encounter a notable challenge: limited transferability. Models trained in one scenario (a.k.a. source domain) often struggle to be effective in another scenario (a.k.a. target domain) [37, 38]. This challenge originates from the heavy reliance on training data that contains unique identities (IDs) like user IDs and item IDs. Since these ID attributes are in general not shareable across scenarios, it impedes the smooth transfer of recommendation models.

Some studies convert sequential recommendation tasks into natural language text and employ In-Context Learning (ICL), which calls OpenAI's API to generate recommendation results [13, 26, 31], reducing the dependence on ID attributes. However, real-world applications reveal that relying solely on ICL may yield inaccurate recommendations due to the overconfidence of LLMs, i.e., the tendency to generate positive predictions [32, 41]. This limitation stems from the disparity between the pre-training corpus of LLMs and the training data of recommendation tasks. Other studies explore parameter-efficient fine-tuning techniques to align LLMs with recommendation tasks in the target domain [2, 38], which exhibit superior performance compared with traditional recommendation models under the few-shot setting. However, cross-domain generalization remains understudied by these approaches.

In this paper, we aim to further improve the performance of LLMs in the target domain by leveraging data from the source domain. Therefore, we propose TLRec, a transfer learning framework designed to enhance the cross-domain generalization of LLMs for

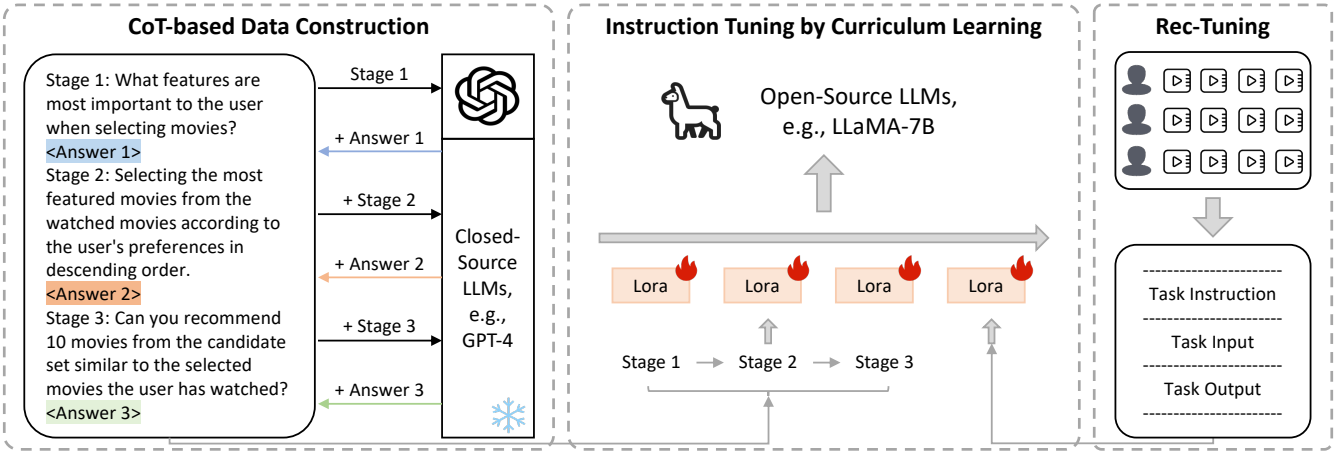


Figure 1: The architecture of TLRec: (i) CoT-based data construction (left). We collect training data from the third-party scenario with the help of OpenAI's GPT-4 API and construct instruction data through CoT. (ii) Instruction tuning by curriculum learning (middle). We employ a curriculum learning strategy to reorder CoT-based data from different stages based on their difficulty and fine-tune LLMs to align with human recommendation thinking. (iii) Rec-tuning (right). To enhance the models' perception of recommendation tasks, we construct recommendation-tuning (rec-tuning) data for further alignment.

sequential recommendation tasks. In our approach, we utilize powerful closed-source LLMs (e.g., GPT-4 [1]) and Chain-of-Thought (CoT) techniques to construct instruction tuning data from the third-party scenario, i.e., the source domain. Effective fine-tuning is ensured through a curriculum learning strategy, refining open-source LLMs (e.g., LLaMA-7B [30]) progressively according to the varying difficulties of instruction tasks. Moreover, comparative experiments demonstrate the superior results of TLRec under the zero-shot and few-shot settings, showcasing its ability to maintain high performance even with limited training data available in the target domain. Our contributions are summarized as follows:

- We propose TLRec, a transfer learning framework that utilizes CoT to construct instruction tuning data from the third-party scenario, augmenting LLMs' performance for sequential recommendation tasks under the zero-shot and few-shot settings.
- To fine-tune LLMs, we adopt curriculum learning, which arranges the instruction tuning data based on difficulty, injecting recommendation knowledge into LLMs more efficiently.
- Comprehensive experiments demonstrate the effectiveness and applicability of TLRec in improving the recommendation performance for LLMs with diverse types and parameter sizes.

2 Related Works

• ***Sequential Recommendation.*** The goal of sequential recommendation is to infer whether the next item aligns with users' preferences based on their historical interactions [6, 20]. While Markov Chain techniques are prevalent in the early attempts [15, 25], deep learning approaches like RNN [9, 11] and Transformer [5, 28] have recently emerged to model user interaction sequences and are becoming mainstream. However, these approaches heavily rely on ID attributes, limiting their transferability across scenarios. Some researchers incorporate natural language features of item titles and item descriptions to improve recommendation systems [13, 26]. Nevertheless, these methods tend to overlook the robust generalization capability of existing LLMs, failing to harness their untapped potential in the recommendation field.

• ***LLM-based Recommendation.*** Recently, there has been growing interest in utilizing LLMs to enhance recommendation systems [33, 40]. One common approach involves representing user behavior as text sequences and using prompts to guide LLMs for recommendation [2, 31, 32, 38]. Moreover, some studies incorporate linguistic information into user/item embeddings for improved performance [41]. In this paper, we focus on directly employing LLMs in recommendation tasks. Previous studies have explored this field, leveraging the interactive capability of LLMs and implementing methods like ICL [2, 13, 31]. For instance, Chat-Rec [13] integrates ChatGPT [4] with traditional recommendation models to build a conversational recommendation system. NIR [31] employs a traditional recommendation model for candidate generation and a multi-step prompting process to re-rank. TALLRec [2] transforms sequential recommendation tasks into the instructional format and constructs training data, aligning LLMs through fine-tuning. However, existing approaches fail to fully exploit the cross-domain generalization ability of LLMs. In contrast, our TLRec introduces transfer learning to enhance LLMs' performance in the target domain by leveraging data from the source domain.

3 Preliminary

In this section, we introduce the preliminary knowledge of instruction tuning for sequential recommendation tasks. Instruction tuning aims to improve LLMs' recommendation performance in the target domain, and the construction of instruction data plays a pivotal role. Specifically, the following four steps are involved: (i) **Task Definition**: Define the task in natural language, denoted as task instructions $\mathcal{S}$. This step involves clearly defining the task and specifying a solution for accomplishment. (ii) **Input/Output**

Table 1: Performance comparison of various methods. The reported metrics are AUC and NDCG@10. The best and second-best results are marked in bold and underlined, respectively. "⋆" denotes statistically significant improvements (i.e., two-sided t-test with $p < 0.05$). All the reported results represent the average of 5 repetitive runs with different random seeds.

	K-shot	SASRec	DROS	DROS-BERT	TALLRec	TLRec
Movie	16	0.5043 / 0.5235	0.5076 / 0.5336	0.5021 / 0.5327	<u>0.6724 / 0.5760</u>	**0.7262⋆ / 0.6038⋆**
	64	0.5048 / 0.5302	0.5154 / 0.5225	0.5171 / 0.5348	<u>0.6748 / 0.5892</u>	**0.7304⋆ / 0.6229⋆**
	256	0.5225 / 0.5519	0.5407 / 0.5660	0.5394 / 0.5663	<u>0.7198 / 0.6002</u>	**0.7411⋆ / 0.6567⋆**
Book	16	0.4948 / 0.4756	0.4928 / 0.4801	0.5007 / 0.4824	<u>0.5636 / 0.5118</u>	**0.6414⋆ / 0.5431⋆**
	64	0.5006 / 0.4883	0.4913 / 0.4889	0.4898 / 0.4991	<u>0.6039 / 0.5220</u>	**0.6426⋆ / 0.5780⋆**
	256	0.5020 / 0.4961	0.4913 / 0.4921	0.5020 / 0.5116	<u>0.6438 / 0.5545</u>	**0.6592⋆ / 0.5997⋆**

Formulation: Formulate inputs and outputs of the task in natural language, which are denoted as task inputs $\mathcal{I}$ and task outputs $\mathcal{O}$, respectively. (iii) **Instruction Data Construction**: Integrate task instructions $\mathcal{S}$ and task inputs $\mathcal{I}$ into instruction inputs $\mathcal{X}$, while use task outputs $\mathcal{O}$ as the corresponding instruction outputs. Then, we define the instruction data as $\mathcal{D} = \{(x, o) \mid x \in \mathcal{X}, o \in \mathcal{O}\}$. (iv) **Instruction Tuning**: Perform instruction tuning for LLMs utilizing the samples in $\mathcal{D}$ as training data.

4 Methodology

In this paper, we propose TLRec, a transfer-learning framework that enhances LLMs for sequential recommendation tasks. Figure 1 illustrates the overall architecture of TLRec.

4.1 CoT-based Data Construction

To improve the recommendation performance of LLMs in the target domain, we aim to construct training data from the third-party scenario for transfer learning. Furthermore, we employ the CoT principle to build instruction data in three stages to ensure effective knowledge injection. These stages align with the cognitive process of human recommendation-making, enabling LLMs to understand sequential recommendation tasks gradually. The three stages are defined as follows: (i) **Stage 1**: Summarize users' behavioral preferences according to given historical interactions. (ii) **Stage 2**: Identify representative items from historical data based on users' preferences. (iii) **Stage 3**: Make recommendations to users from the candidate set with reference to identified items.

Figure 1 shows the design of task inputs for each stage, with GPT-4 API used to obtain the corresponding task outputs. In our approach, we create a dedicated candidate set for each target user to limit the recommendation options. Items from the candidate set are used to generate task outputs in Stage 3. For simplicity, we adopt a user-filtering approach to construct the candidate set [31]. Notably, the construction of the candidate set can vary depending on specific requirements. When comparing traditional single-stage instruction data (like the format mentioned in [2]) with our three-stage instruction data, it becomes clear that the single-stage data is inadequate for effectively teaching LLMs to understand users' interests. On the other hand, our proposed three-stage instruction data successfully addresses this limitation by accomplishing three distinct subtasks: capturing users' preferences, ranking previously interacted items based on the preferences, and recommending the most similar items from the candidate set, simulating the thinking process of humans when recommending items to their friends.

4.2 Instruction Tuning by Curriculum Learning

The obtained CoT-based data is divided into three stages. We avoid directly fine-tuning LLMs with them. Directly fine-tuning without considering the difficulty may result in inferior performance, as LLMs struggle to extract valid information from the mixed-difficulty data. In our experiments, we provide comparative results to support this finding. To address this challenge, we employ a curriculum learning strategy, which helps LLMs effectively learn from the data. Specifically, we arrange the data in order of increasing difficulty (Stage 1 to Stage 3) and combine them as the final instruction dataset $\mathcal{D}$. The dataset is treated as the training set for model fine-tuning. This approach resembles gradually teaching complex concepts to a young child, ensuring better comprehension.

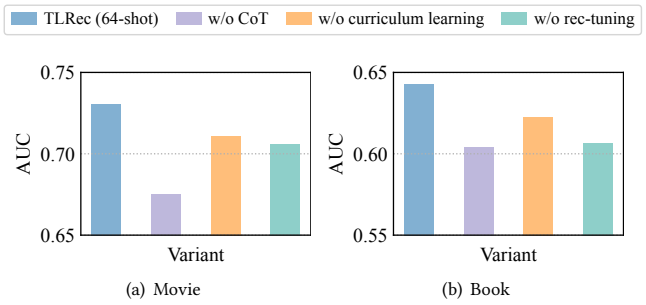
Fine-tuning entire LLMs demands significant computational resources. Therefore, we employ Low-Rank Adaptation (LoRA) [18], which leverages the concept that language models contain numerous parameters but with crucial information concentrated in lower-dimensional spaces. By fine-tuning only a small subset of parameters, we achieve comparable performance. This entails freezing pre-trained model parameters and introducing trainable rank decomposition matrices. The objective is computed as follows:

$$\max_{\Phi} \sum_{(x,o)\in\mathcal{D}} \sum_{t=1}^{|o|} \log(P(o_t \mid x, o_{<t}; \Theta + \Phi)), \tag{1}$$

where Θ is the original LLM parameters and Φ is the LoRA parameters. o_t represents the t-th token of task output $o \in \mathcal{O}$, and $o_{<t}$ represents the preceding tokens. This approach strikes a balance between model performance and computational efficiency.

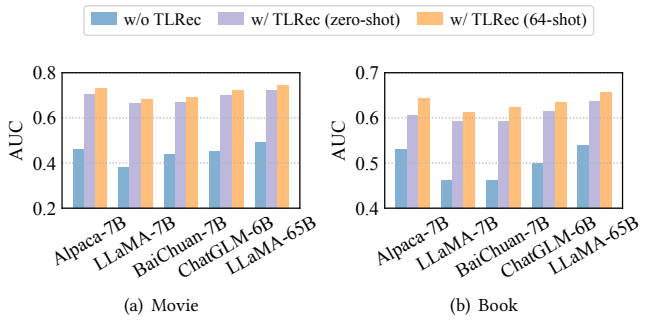
4.3 Rec-tuning

Following the previous steps, LLMs gain the basic recommendation capability. To further improve the performance, we employ a rec-tuning process, which utilizes task-specific data. The data format for rec-tuning differs from the previous CoT format. We convert task-specific data into the format described in [2] and fine-tune LLMs. The data consists of two parts: from the third-party scenario and from the target scenario, representing the zero-shot and few-shot settings in sequential recommendation tasks. Under the zero-shot setting, our goal is to generate recommendations for a target scenario without any training data, i.e., utilizing training data only from the third-party scenario. LLMs are fine-tuned using third-party data and evaluated on the target domain data. Moreover, under the



(a) Movie (b) Book

Figure 2: Results of the ablation study on Movie and Book.



(a) Movie (b) Book

Figure 3: Performance of TLRec (zero-shot) and TLRec (64-shot) using LLMs with diverse types and parameter sizes.

few-shot setting, we additionally use a few pieces of data from the target scenario for further alignment.

5 Experiments

In this section, we conduct experiments with the aim of answering the following research questions:

- RQ1: How does TLRec perform compared with baselines?
- RQ2: What is the role of each designed module in TLRec?
- RQ3: What is the performance of TLRec when using LLMs of different types and parameter sizes?

5.1 Experimental Setup

5.1.1 ***Datasets.*** We conduct experiments on two sequential recommendation datasets. (i) **Movie**: This dataset is processed from MovieLens-100K [14], which consists of user ratings (1∼5) on movies and textual descriptions like title and publish year. (ii) **Book**: This dataset is processed from BookCrossing [45], containing user ratings (1∼10) and textual descriptions of books, such as title and author. We follow the approach used in previous studies [2, 39] by considering 10 interacted items per user as historical interactions. The datasets are divided into training, validation, and test sets in a ratio of 8:1:1. Additionally, interactions with ratings > 3 are treated as likes and others as dislikes on Movie, while the threshold is set to 5 on Book. The Netflix [3] and Amazon-Book datasets [16] are introduced as third-party scenarios to construct CoT-based data. Under the zero-shot setting, we solely utilize third-party data for fine-tuning. Moreover, we randomly choose K samples for few-shot training, where $K \in \{16, 64, 256\}$, assessing the performance of various methods trained on limited target domain data.

5.1.2 ***Baselines.*** We compare TLRec with the following baselines: (i) **SASRec**: A standard Transformer-based recommendation method [19]. (ii) **DROS**: A method that employs distributionally robust optimization for recommendation [35]. (iii) **DROS-BERT**: A variant of DROS which incorporates a pre-trained BERT model as the encoder [10]. (iv) **TALLRec**: A framework that aligns LLMs with sequential recommendation tasks through instruction tuning [2].

5.1.3 ***Evaluation and Implementation Details.*** We adopt AUC and NDCG@10 to evaluate the recommendation performance. Specifically, NDCG@10 is a position-aware metric that gives larger weights to higher positions. Note that as we have only one test item per user, for each user u, we randomly sample 50 negative items and rank them along with the ground-truth item. Based on the rankings of these 51 items, NDCG@10 can be evaluated. To construct CoT-based data, we call OpenAI's GPT-4 API to gather responses for diverse instruction tasks, with the hyperparameter temperature set to 0.2. Similar to [2], we employ Alpaca-7B [29], a model fine-tuned from LLaMA-7B [30], as the recommendation backbone. LoRA [18] is used for instruction tuning with a rank of 8. We use the AdamW optimizer with a learning rate of 1e-4 and incorporate a linear warmup for adjustment. More experimental details are presented in Appendix A, while Appendix B showcases examples of fine-tuning data including CoT-based data and rec-tuning data.

5.2 Performance Comparison (RQ1)

Table 1 presents the performance comparison of TLRec and other baselines on two datasets. From the experimental results, we make the following observations: (i) TLRec achieves significant improvements in both AUC and NDCG@10 compared with baselines, harnessing abundant knowledge resources within LLMs for sequential recommendation tasks. Additionally, TLRec outperforms TALLRec, highlighting the effectiveness of CoT and curriculum learning during instruction tuning on the third-party dataset, i.e., data from the source domain can improve LLMs' recommendation performance in the target domain. (ii) Except for TALLRec, baselines consistently show worse performance under the few-shot setting (AUC is close to 0.5), indicating their limited capacity to rapidly acquire recommendation skills with a small training dataset.

5.3 Ablation Study (RQ2)

In this part, we conduct an ablation study to explore the contribution of each designed module in TLRec. We compare TLRec with three variants: (i) w/o CoT, where not using the third-party data to enhance LLMs, (ii) w/o curriculum learning, i.e., using CoT-based data without rearranging, and (iii) w/o rec-tuning, which excludes the downstream recommendation data for further alignment. Figure 2 illustrates the recommendation performance of different variants on two datasets. It is evident that all the designed modules play a crucial role in TLRec. Removing any module leads to a deterioration in performance. For example, as shown in Figure 2(a) and 2(b), each variant exhibits inferior AUC. The CoT-based data is utilized to align natural language with recommendation tasks, injecting knowledge into LLMs. The rec-tuning data from third-party scenarios follows the same format as downstream tasks, enabling LLMs to learn the task format of the target domain. Lastly, a small downstream task dataset is employed for few-shot fine-tuning.

5.4 Applicability Discussion (RQ3)

TLRec is compatible with a wide range of open-source LLMs. To highlight the versatility of our framework, we evaluate it using five open-source LLMs of different parameter sizes: (i) Alpaca-7B [29], (ii) LLaMA-7B [30], (iii) BaiChuan-7B [34], (iv) ChatGLM-6B [12], and (v) LLaMA-65B [30]. Notably, TLRec can fully enhance the recommendation performance of LLMs under both zero-shot and few-shot settings. Thus, we employ TLRec (zero-shot) and TLRec (64-shot) as plug-and-play adapters, respectively. The results in Figure 3 demonstrate two main findings: (i) Utilizing the TLRec framework with different LLMs consistently enhances recommendation performance. (ii) LLMs with larger parameter sizes generally perform better for sequential recommendation (e.g., LLaMA-65B vs. LLaMA-7B). These results highlight TLRec's potential to leverage third-party data for enhancing LLMs' recommendation performance in the target domain.

6 Conclusion

This paper proposes TLRec, a transfer learning framework that improves LLMs' performance for sequential recommendation tasks through CoT-based data construction and instruction tuning by curriculum learning, efficiently leveraging the third-party data to achieve better cross-domain generalization. Our experimental results validate the effectiveness of TLRec, showcasing significant improvement over traditional recommendation models.

A More Experimental Details

In the experiments, we treat Movie[1] and Book[2] as the target domain and introduce Netflix[3] and Amazon-Book[4] as the third-party scenario, i.e., the source domain. We generate CoT-based data and rec-tuning data from the source domain, transferring knowledge to enhance LLMs' recommendation performance in the target domain through instruction tuning, for better cross-domain generalization.

B Illustrations of Fine-tuning Data

B.1 CoT-based Data

To effectively utilize third-party data to improve LLMs' performance for sequence recommendation tasks in the target domain, we call OpenAI's GPT-4 API to collect training data and apply the CoT principle to divide the data into three stages. Inspired by curriculum learning, we arrange these stages by difficulty and combine them for model fine-tuning. Table 2 shows an example of CoT-based data, formatted for instruction tuning. We start by specifying a task definition as the task instruction. The goal of CoT-based data is to inject recommendation knowledge into LLMs, enabling them to understand sequential recommendation tasks. Therefore, for the task input, users' preferences are summarized and representative items are selected, i.e., answers to Stage 1 and Stage 2. Then, we define the task output as the answer to Stage 3. These answers are all generated by GPT-4. This approach guides LLMs to simulate the human recommendation process. We further filter CoT-based data to include only correct recommendations, avoiding training noise.

[1]https://grouplens.org/datasets/movielens
[2]http://www2.informatik.uni-freiburg.de/~cziegler/BX
[3]https://www.kaggle.com/datasets/netflix-inc/netflix-prize-data
[4]https://www.kaggle.com/datasets/mohamedbakhet/amazon-books-reviews

Table 2: An example of CoT-based data.

Task Instruction:
Given the candidate set, the user's watched movies, the user's preferences, and the most featured movies, recommend 10 movies from the candidate set that are similar to the selected movies the user has watched.
Task Input:
Candidate Set: "Napoleon Dynamite (2004)", "50 First Dates (2004)", "Top Secret! (1984)", · · · . Watched Movies: "The Final Cut (2004)", "Finding Neverland (2004)", "This Is Spinal Tap (1984)", · · · . Stage 1: What features are most important to the user when selecting movies? <Answer 1>: The user seems to enjoy some genres, such as comedy, drama, adventure, and musicals. · · · . Stage 2: Selecting the most featured movies from the watched movies according to the user's preferences in descending order. <Answer 2>: 1. "This Is Spinal Tap (1984)", 2. "History of the World: Part 1 (1981)", 3. "Finding Neverland (2004)", · · · . Stage 3: Can you recommend 10 movies from the candidate set similar to the selected movies the user has watched?
Task Output:
<Answer 3>: 1. "This Is Spinal Tap (1984)" - "Top Secret! (1984)", 2. "Finding Neverland (2004)" - "50 First Dates (2004)", · · · .

Table 3: An example of rec-tuning data.

Task Instruction:
Given the user's preferences and unpreferences, identify whether the user will like the target movie by answering "Yes." or "No.".
Task Input:
User Preferences: "A Beautiful Mind (2001)", · · · . User Unpreferences: "This Is Spinal Tap (1984)", · · · . Whether the user will like the target movie: "The King and I (1956)"?
Task Output:
No.

B.2 Rec-tuning Data

Different from CoT-based data, rec-tuning data is used to guide LLMs on how to directly employ them for sequence recommendation tasks. As shown in Table 3, we instruct LLMs to determine whether the next item meets users' interests based on historical interactions. For task input, we concatenate user preferences, user unpreferences, and the target item. To formulate the task output, we convert the recommendation task into a binary decision-making process, i.e., answering "Yes." or "No.". Notably, under the zero-shot setting, we use rec-tuning data from the source domain, while under the few-shot setting, we additionally use a small rec-tuning dataset from the target domain.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, and Shyamal Anadkat. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] 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 *ACM Conference on Recommender Systems (RecSys)*. 1007–1014.

[3] James Bennett and Stan Lanning. 2007. The netflix prize. In *Proceedings of KDD Cup and Workshop*. 1–4.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, and Amanda Askell. 2020. Language models are few-shot learners. *Conference on Neural Information Processing Systems (NeurIPS)* (2020), 1877–1901.

[5] Qiwei Chen, Changhua Pei, Shanshan Lv, Chao Li, Junfeng Ge, and Wenwu Ou. 2021. End-to-end user behavior retrieval in click-through rateprediction model. *arXiv preprint arXiv:2108.04468* (2021).

[6] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Deep Learning Practice for High-Dimensional Sparse Data with KDD (DLP-KDD)*. 1–4.

[7] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, and Sebastian Gehrmann. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research (JMLR)* (2023), 1–113.

[8] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, and Siddhartha Brahma. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research (JMLR)* (2024), 1–53.

[9] Qiang Cui, Shu Wu, Qiang Liu, Wen Zhong, and Liang Wang. 2018. MV-RNN: A multi-view recurrent neural network for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* (2018), 317–331.

[10] 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).

[11] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. 2017. Sequential user-based recurrent neural network recommendations. In *ACM Conference on Recommender Systems (RecSys)*. 152–160.

[12] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2021. Glm: General language model pretraining with autoregressive blank infilling. *arXiv preprint arXiv:2103.10360* (2021).

[13] 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* (2023).

[14] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems* (2015), 1–19.

[15] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *IEEE International Conference on Data Mining (ICDM)*. 191–200.

[16] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *International World Wide Web Conference (WWW)*. 507–517.

[17] Wanwei He, Yinpei Dai, Binyuan Hui, Min Yang, Zheng Cao, Jianbo Dong, Fei Huang, Luo Si, and Yongbin Li. 2022. SPACE-2: Tree-Structured Semi-Supervised Contrastive Pre-training for Task-Oriented Dialog Understanding. In *International Conference on Computational Linguistics (COLING)*. 553–569.

[18] 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* (2021).

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

[20] Yang Li, Tong Chen, Peng-Fei Zhang, and Hongzhi Yin. 2021. Lightweight self-attentive sequential recommendation. In *ACM International Conference on Information and Knowledge Management (CIKM)*. 967–977.

[21] Jiaye Lin, Qing Li, Guorui Xie, Zhongxu Guan, Yong Jiang, Ting Xu, Zhong Zhang, and Peilin Zhao. 2024. Mitigating Sample Selection Bias with Robust Domain Adaption in Multimedia Recommendation. In *ACM International Conference on Multimedia (MM)*. 1–10.

[22] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2024. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Conference on Neural Information Processing Systems (NeurIPS)* (2024), 1–15.

[23] Ansong Ni, Srini Iyer, Dragomir Radev, Veselin Stoyanov, Wen-tau Yih, Sida Wang, and Xi Victoria Lin. 2023. Lever: Learning to verify language-to-code generation with execution. In *International Conference on Machine Learning (ICML)*. 26106–26128.

[24] Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *ACM International Conference on Information and Knowledge Management (CIKM)*. 2685–2692.

[25] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *International World Wide Web Conference (WWW)*. 811–820.

[26] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. In *Conference on Neural Information Processing Systems (NeurIPS)*. 1–13.

[27] Murray Shanahan, Kyle McDonell, and Laria Reynolds. 2023. Role play with large language models. *Nature* (2023), 493–498.

[28] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *ACM International Conference on Information and Knowledge Management (CIKM)*. 1441–1450.

[29] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research* (2023), 7.

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

[31] Lei Wang and Ee-Peng Lim. 2023. Zero-shot next-item recommendation using large pretrained language models. *arXiv preprint arXiv:2304.03153* (2023).

[32] 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 *ACM International Conference on Web Search and Data Mining (WSDM)*. 806–815.

[33] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, and Qi Liu. 2023. A survey on large language models for recommendation. *arXiv preprint arXiv:2305.19860* (2023).

[34] Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, and Dong Yan. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305* (2023).

[35] Zhengyi Yang, Xiangnan He, Jizhi Zhang, Jiancan Wu, Xin Xin, Jiawei Chen, and Xiang Wang. 2023. A generic learning framework for sequential recommendation with distribution shifts. In *International SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 331–340.

[36] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Conference on Neural Information Processing Systems (NeurIPS)* (2024), 1–14.

[37] 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 *International SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 2639–2649.

[38] 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).

[39] Yang Zhang, Tianhao Shi, Fuli Feng, Wenjie Wang, Dingxian Wang, Xiangnan He, and Yongdong Zhang. 2023. Reformulating CTR Prediction: Learning Invariant Feature Interactions for Recommendation. In *International SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1386–1395.

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

[41] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Adapting large language models by integrating collaborative semantics for recommendation. *arXiv preprint arXiv:2311.09049* (2023).

[42] Qinkai Zheng, Xiao Xia, Xu Zou, Yuxiao Dong, Shan Wang, Yufei Xue, Zihan Wang, Lei Shen, Andi Wang, and Yang Li. 2023. Codegeex: A pre-trained model for code generation with multilingual evaluations on humaneval-x. *arXiv preprint arXiv:2303.17568* (2023).

[43] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *AAAI Conference on Artificial Intelligence (AAAI)*. 5941–5948.

[44] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD)*. 1059–1068.

[45] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *International World Wide Web Conference (WWW)*. 22–32.