TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation

Keqin Bao*
baokq@mail.ustc.edu.cn
University of Science and Technology of China
China

Jizhi Zhang*
cdzhangjizhi@mail.ustc.edu.cn
University of Science and Technology of China
China

Yang Zhang
zy2015@mail.ustc.edu.cn
University of Science and Technology of China
China

Wenjie Wang
wenjiewang96@gmail.com
National University of Singapore
Singapore

Fuli Feng†
fulifeng93@gmail.com
University of Science and Technology of China
China

Xiangnan He†
xiangnanhe@gmail.com
University of Science and Technology of China
China

ABSTRACT

Large Language Models (LLMs) have demonstrated remarkable performance across diverse domains, thereby prompting researchers to explore their potential for use in recommendation systems. Initial attempts have leveraged the exceptional capabilities of LLMs, such as rich knowledge and strong generalization through In-context Learning, which involves phrasing the recommendation task as prompts. Nevertheless, the performance of LLMs in recommendation tasks remains suboptimal due to a substantial disparity between the training tasks for LLMs and recommendation tasks, as well as inadequate recommendation data during pre-training. To bridge the gap, we consider building a Large Recommendation Language Model by tunning LLMs with recommendation data. To this end, we propose an efficient and effective Tuning framework for Aligning LLMs with Recommendations, namely TALLRec. We have demonstrated that the proposed TALLRec framework can significantly enhance the recommendation capabilities of LLMs in the movie and book domains, even with a limited dataset of fewer than 100 samples. Additionally, the proposed framework is highly efficient and can be executed on a single RTX 3090 with LLaMA-7B. Furthermore, the fine-tuned LLM exhibits robust cross-domain generalization. Our code and data are available at https://github.com/SAI990323/TALLRec.

CCS CONCEPTS

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

KEYWORDS

Recommendation, Instruction Tuning, Large Language Models

*The two authors contributed equally to this work and are listed alphabetically.
†The corresponding author.

RecSys '23, September 18–22, 2023, Singapore, Singapore

ACM ISBN 979-8-4007-0241-9/23/09…$15.00
https://doi.org/10.1145/3604915.3608857

ACM Reference Format:
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 *Seventeenth ACM Conference on Recommender Systems (RecSys '23), September 18–22, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3604915.3608857

1 INTRODUCTION

Large Language Models (LLMs) have exhibited remarkable proficiency in generating text that closely resembles human language and in performing a wide range of tasks [69], including Natural Language Processing [4, 32, 54], Robotics [10, 43, 56], and Information Retrieval [1, 24, 25, 48]. Prior research has also demonstrated the knowledge-rich and compositional generalization capabilities of LLMs [36, 41, 53]. Only given appropriate instructions, these models are able to learn how to solve unseen tasks and inspire their own knowledge to achieve a high level of performance [33]. The aforementioned capabilities of LLM present promising opportunities to address the current challenges requiring strong generalization and rich knowledge in the recommendation field. In this light, it is valuable to explore the integration of LLMs into recommender systems, which has received limited attention in prior research.

In recent initial attempts [13, 47], achieving the target relies on *In-context Learning* [3], which is typically implemented through the official OpenAI API [2]. They regard the LLM as a toolformer [42] of traditional recommendation models (such as MF [27] and LightGCN [16]), *i.e.,* the LLM is used for re-ranking the candidate items filtered by these models. However, these approaches only reach a comparable performance with traditional models [13, 47]. Worse still, using only In-context Learning may fail to make recommendations. As shown in Figure 1, we find that ChatGPT either refuses to answer or always gives positive predictions (likes). Therefore, it is critical to further explore an appropriate way for more effective leverage of LLMs in the recommendation.

We postulate that the failure of using only In-context Learning is because of two reasons: 1) LLMs may not align well with the recommendation task due to the huge gap between language processing tasks for training LLMs and recommendation. Besides, the recommendation-oriented corpus is very limited during the

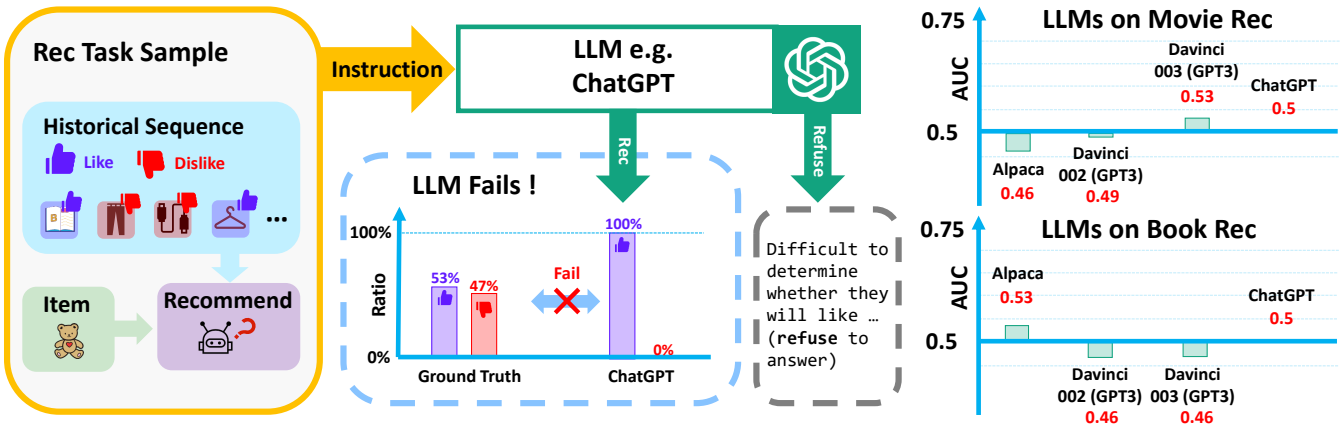

Figure 1: Illustration of LLMs for the recommendations. Given users' interaction history, LLMs predict whether a user will like a new item through In-context Learning. However, the representative LLMs, *e.g.,* ChatGPT, either refuse to answer or always give positive predictions (likes) on Movie and Book recommendation tasks. If we ignore the refused answers and calculate AUC on the remaining samples, we find that LLMs perform similarly with random guessing (AUC=0.5). Refer to Section 3 for more experimental details.

training phase of LLMs. 2) The effect of LLMs is restricted by the underlying recommendation models, which may fail to include target items in their candidate lists due to their limited capacity. Therefore, we consider building a *Large Recommendation Language Model* (LRLM) to bridge the gap between LLMs and the recommendation task and better stimulate the recommendation capabilities of LLMs in addition to In-context Learning.

Toward this goal, we focus on tuning LLMs with the recommendation task. Considering that instruction tuning is core to letting the LLM learn to solve different tasks and have strong generalization ability [22, 23, 37], we propose a lightweight tuning framework to adapt LLMs for recommendations, named TALLRec. Elaborately, TALLRec structures the recommendation data as instructions and tunes the LLM via an additional instruction tuning process. Moreover, given that LLM training necessitates a substantial amount of computing resources, TALLRec employs a lightweight tuning approach to efficiently adapt the LLMs to the recommendation task.

Specifically, we apply the TALLRec framework on the LLaMA-7B model [46] with a LoRA [21] architecture, which ensures the framework can be deployed on an Nvidia RTX 3090 (24GB) GPU. Furthermore, to investigate the minimal computational resources required, we do experiments in a few-shot setting, utilizing only a limited number of tuning examples. We conduct detailed experiments in knowledge-rich recommendation scenarios of movies and books, where the tuned LLaMA-7B model outperforms traditional recommendation models and In-context Learning with GPT3.5, a much stronger LLM than LLaMA-7B. The results validate the efficiency and robustness of our framework: 1) our TALLRec framework can quickly inspire the recommendation capability of LLMs in the few-shot setting. and 2) LLMs trained via the TALLRec framework have a strong generalization ability across different domains (*e.g., movie* $\rightarrow$ *book*).

In total, our contributions are summarized as follows:

- We study a new problem in recommendation — aligning the LLMs with the recommendation, where we reveal the limitations of In-context Learning-based approaches and underscore the significance of instruction tuning.
- We introduce a new TALLRec framework to build Large Recommendation Language Models, which enables the effective and efficient tuning of LLMs for recommendation with low GPU costs and few tuning samples.
- We conduct extensive experiments, validating the effectiveness and efficiency of the proposed framework, and uncovering its exceptional robustness with seamless navigation across different domains.

2 TALLREC

In this section, we first introduce the preliminary knowledge for tuning LLMs and our task formulation, and then present the proposed TALLRec framework.

2.1 Preliminary

Table 1: A tuning sample for a translation task.

Instruction Input	
Task Instruction:	Translate from English to Chinese.
Task Input:	Who am I ?
Instruction Output	
Task Output:	我是谁?

• **Instruction Tuning** is a crucial technique to train LLMs with human-annotated instructions and responses [36]. Generally, instruction tuning has four steps (see the example in Table 1). Specifically, **Step 1:** Define a task and articulate a "*Task Instruction*" using natural language, which usually encompasses a clear definition of the task, as well as specific solutions to address it. **Step 2:** Formulate and construct the input and output of the task in natural language, denoted as "*Task Input*" and "*Task Output*". **Step 3:** Integrate the "*Task Instruction*" and "*Task Input*" together to form the "*Instruction*

Table 2: A tuning sample for rec-tuning.

Instruction Input	
Task Instruction:	Given the user's historical interactions, please determine whether the user will enjoy the target new movie by answering "Yes" or "No".
Task Input:	User's liked items: GodFather. User's disliked items: Star Wars. Target new movie: Iron Man
Instruction Output	
Task Output:	No.

Input", and take the "*Task Output*" as the corresponding "*Instruction Output*", for each tuning sample. **Step 4:** Instruction tuning on LLMs based on the formatted pairs of "*Instruction Input*" and "*Instruction Output*".

• **Rec-tuning Task Formulation.** We aim to utilize LLM, denoted as $\mathcal{M}$, to construct an LRLM, which can predict whether a new item will be enjoyed by a user. To achieve this objective, we do recommendation tuning (rec-tuning) on LLMs with recommendation data. As shown in Table 2, we format recommendation data into a pattern of instruction tuning. We begin by composing a "*Task Instruction*" that directs the model to determine whether the user will like the target item based on their historical interactions, and to respond with a binary answer of "Yes" or "No". To format the "*Task Input*", we categorize the user's historically interacted items into two groups based on ratings: the user's liked items and disliked items, where items are sequentially ranked by interaction time and represented by textual descriptions (*e.g.,* title and brief introduction). Besides, "*Task Input*" also includes a target new item that the user has never seen. Lastly, we merge "*Task Instruction*" and "*Task Input*" to create a "*Instruction Input*", and then set the expected "*Instruction Output*" as 'Yes" or "No" for rec-tuning.

2.2 TALLRec Framework

In this subsection, we introduce the TALLRec framework, which aims to facilitate the effective and efficient alignment of LLMs with recommendation tasks, particularly in low GPU memory consumption settings. Specifically, we first present two TALLRec tuning stages with lightweight implementation, followed by the backbone selection. As shown in Figure 2, TALLRec comprises two tuning stages: alpaca tuning and rec-tuning. The former stage is the common training process of LLM that enhances LLM's generalization ability, while the latter stage emulates the pattern of instruction tuning and tunes LLMs for the recommendation task.

• **TALLRec Tuning Stages.** For alpaca tuning, we employ the self-instruct data made available by Alpaca [45] to train the LLM. Specifically, we utilize the conditional language modeling objective during the alpaca tuning, as exemplified in the Alpaca repository[1]. Formally,

$$\max_{\Phi} \sum_{(x,y)\in\mathcal{Z}} \sum_{t=1}^{|y|} \log\left(P_{\Phi}(y_t|x, y_{<t})\right), \tag{1}$$

where x and y represent the "*Instruction Input*" and "*Instruction Output*" in the self-instruct data, respectively, y_t is the t-th token of the y, $y_{<t}$ represents the tokens before y_t, Φ is the original parameters of $\mathcal{M}$, and $\mathcal{Z}$ is the training set. For rec-tuning, we can leverage the rec-tuning sample as described in Table 2 to tune the LLM, similar to alpaca tuning.

• **Lightweight Tuning.** However, directly tuning the LLM is computationally intensive and time-consuming. As such, we propose to adopt a lightweight tuning strategy to execute both alpaca tuning and rec-tuning. The central premise of lightweight tuning is that contemporary language models may possess an excessive number of parameters, and their information is concentrated on a low intrinsic dimension [21]. Consequently, we can achieve comparable performance to that of the entire model by tuning only a small subset of parameters [20, 28, 31]. Specifically, we employ LoRA [21], which involves freezing the pre-trained model parameters and introducing trainable rank decomposition matrices into each layer of the Transformer architecture to facilitate lightweight tuning. Therefore, by optimizing rank decomposition matrices, we can efficiently incorporate supplementary information while maintaining the original parameters in a frozen state. In total, the final learning objective can be computed as:

$$\max_{\Theta} \sum_{(x,y)\in\mathcal{Z}} \sum_{t=1}^{|y|} \log\left(P_{\Phi+\Theta}(y_t|x, y_{<t})\right), \tag{2}$$

where Θ is the LoRA parameters and we only update LoRA parameters during the training process. Through LoRA, we can complete training with only one-thousandth of the original LLM parameters to complete the training process [21]. ·

• **Backbone Selection**. At present, there are large amounts of LLMs released, such as GPT series, PaLM, CHinchilla, and LLaMA [3, 4, 18, 46]. Among these, a considerable number of LLMs (such as PaLM and Chinchilla) do not provide access to their model parameters or APIs, rendering them challenging to utilize for research or other applications. Additionally, data security concerns are significant issues in the recommendation field. Consequently, the utilization of third-party APIs (such as ChatGPT and text-davinci-003) to leverage LLMs necessitates further discussion. To replicate the issues that require consideration in real-world recommendation scenarios, we intend to simulate the practical utilization of a public LLM and update its parameters for recommendation purposes. After careful consideration, we have opted to conduct experiments using LLMs-LLaMA, which is presently the best-performing open-source LLM, and whose training data is also publicly available [46].

3 EXPERIMENTS

In this section, we conduct experiments to answer the following research questions:

- **RQ1:** How does TALLRec perform compared with current LLM-based and traditional recommendation models?
- **RQ2:** How do the different components in TALLRec affect its effectiveness?
- **RQ3:** How does TALLRec perform under cross-domain recommendation?

• **Dataset.** We conduct experiments on two datasets. The statistics and more details can be found in our released data.

[1] https://github.com/tloen/alpaca-lora.

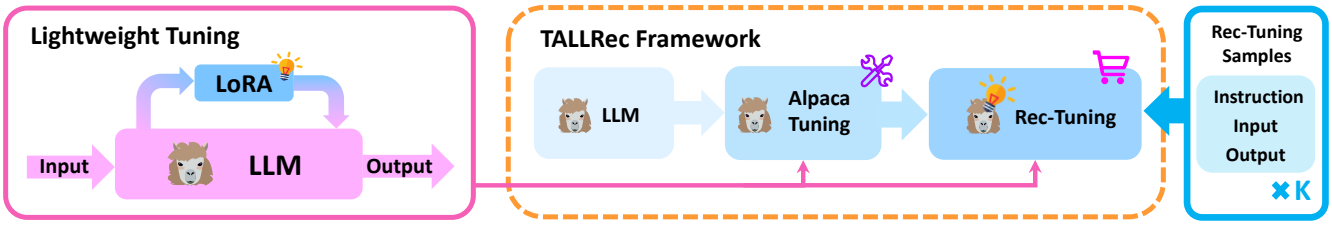

Figure 2: Illustration of the TALLRec framework constructed by alpaca tuning and rec-tuning two stages. During rec-tuning, we use the rec-tuning samples with instruction input and output constructed from recommendation data. Notably, we employ lightweight tuning technology to enhance the efficiency of our TALLRec framework.

- **Movie.** This is a processed dataset from MovieLens100K [12] , which comprises user ratings on movies and comprehensive textual descriptions of movies such as "title" and "director". Because we conduct experiments in a few-shot training setting that requires limited tuning samples, we process the original dataset by sampling the most recent 10,000 interactions and split them into training, validation, and testing sets with a ratio of 8:1:1. To construct a rec-tuning sample, 10 interactions prior to the target item are retained as historical interactions. Following [16, 66], we only treat the interactions with ratings > 3 as "likes", and those with ratings ≤ 3 as "dislike".
- **Book.** This is a book recommendation dataset processed from BookCrossing [71]. The BookCrossing dataset has user ratings (1-10) and textual descriptions of books, such as the information of 'Book-Author' and 'Book-Title'. For each user, we randomly select an item interacted by this user as the prediction target, and sample 10 interacted items as historical interactions[2]. Subsequently, we partition constructed rec-tuning samples into training, validation, and testing sets with the same ratio of 8:1:1. Additionally, we binarize the ratings according to a threshold of 5.

• **Few-shot Training Setting.** We adopt a few-shot training setting, where only a limited number of samples are randomly selected from the training set for model training. It is referred to as 'K-shot' training setting, where K represents the number of training samples used. By setting an extremely small value for K, such as 64, we could test whether a method can rapidly acquire recommendation capability from LLMs with severely limited training data.

• **Baseline.** We compare TALLRec against both LLM-based and traditional recommendation methods. 1) Existing **LLM-based methods** adopt In-context Learning to directly generate recommendations [13, 47]. For a fair comparison, we align these methods with TALLRec by using the same instructions. Specifically, we perform In-context Lerning on different LLMs: 1) *Alpaca-LoRA*, 2) *Text-Davinvi-002*, 3) *Text-Davinviv-003*, and 4) *ChatGPT*. Alpaca-LoRA is a model for reproducing Alpaca results of the LLaMA model by using LoRA and alpaca tuning. The latter three are GPT series models from OpenAI.

2) **Traditional methods.** Since our approach utilizes historical interactions to predict the subsequent interaction, similar to the sequential recommendation, we consider the following sequential models: **(i) GRU4Rec [17]** is an RNN-based sequential recommender, which utilizes GRU to encode historical interactions. **(ii) Caser [44]** utilizes CNN to encode historical interaction sequences. **(iii) SASRec [26]** is a classic transformer-based sequential recommender. **(iv) DROS [60]** is a state-of-the-art sequential recommender model, which harnesses distributionally robust optimization for robust recommendations. We use the version implemented by GRU4Rec, provided by the authors.[3] The sequential models above rely on item ID features without considering textual descriptions of items. However, in our setting, we assume item text descriptions are available for LLM tuning. To ensure fair comparisons, we further consider comparing the following variants of GRU4Rec and DROS: **(v) GRU-BERT** is a variant of GRU4Rec that incorporates a pre-trained BERT [7] to encode text descriptions. Specifically, BERT encodes text descriptions and outputs a CLS embedding, which is then concatenated with the original ID embeddings of GRU4Rec as the item representations. **(vi) DROS-BERT** is integrated with BERT, similar to GRU-BERT.

• **Evaluation Metric.** Since TALLRec aims to predict user preference over a given target item, *i.e.,* a binary classification problem, we adopt a popular evaluation metric used in recommendation: Area Under the Receiver Operating Characteristic (AUC).

• **Implementation Details.** To ensure uniform sequence lengths, we use the user's last interacted item to pad the historical interaction sequences with lengths $<$ the threshold, 10. For all methods, we optimize parameters using Adam with MSE loss and a learning rate of 1e-3. We search the weight decay of all methods in {1e-3, 1e-4, 1e-5, 1e-6, 1e-7}. Following [60], regarding specific hyperparameters of baselines, we adhered to their original settings. For GRU-BERT and DROS-BERT, we utilize BERT released by Hugging Face[4], while setting the number of GRU layers to 4 and the hidden size to 1024 for aligning with BERT's embedding size. Lastly, we run all methods five times with different random seeds and report the averaged results.

3.1 Performance Comparison (RQ1)

We aim to investigate the recommendation performance of various methods under the few-shot training setting, which enables us to evaluate how LLMs can be quickly adjusted for recommendation with limited rec-tuning samples. The evaluation results against traditional methods are presented in Table 3, while the comparison against LLM-based methods is depicted in Figure 3 (a).

[2]BookCrossing lacks interaction timestamps, thus we can only construct historical interaction by random sampling.

[3]https://github.com/YangZhengyi98/DROS.

[4]https://huggingface.co/bert-base-uncased.

Table 3: Performance comparison between conventional sequential recommendation baselines and TALLRec under different few-shot training settings. The reported result is the AUC multiplied by 100, with boldface indicating the highest score. ‡: significantly better than all baselines with t-test p<0.01.

	Few-shot	GRU4Rec	Caser	SASRec	DROS	GRU-BERT	DROS-BERT	TALLRec
movie	16	49.07	49.68	50.43	50.76	50.85	50.21	**67.24‡**
	64	49.87	51.06	50.48	51.54	51.65	51.71	**67.48‡**
	256	52.89	54.20	52.25	54.07	53.44	53.94	**71.98‡**
book	16	48.95	49.84	49.48	49.28	50.07	50.07	**56.36**
	64	49.64	49.72	50.06	49.13	49.64	48.98	**60.39‡**
	256	49.86	49.57	50.20	49.13	49.79	50.20	**64.38‡**

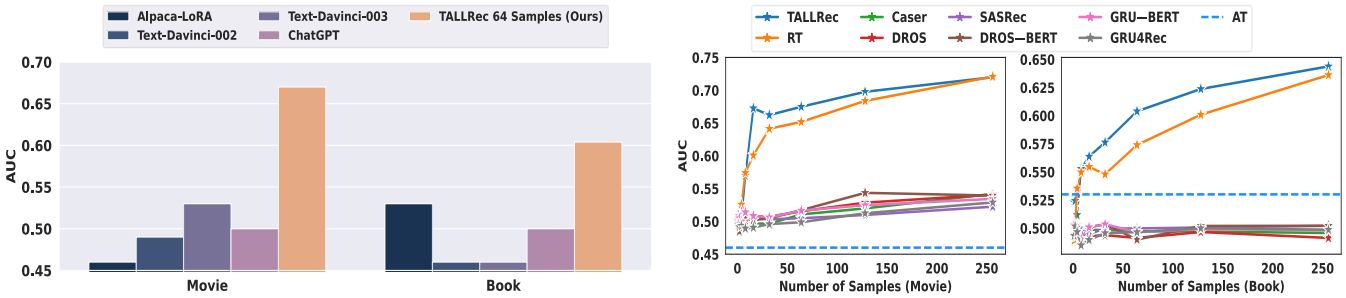

Figure 3: Figure (a) shows the performance comparison between LLM-based baselines (zero-shot setting) and ours TALLRec, where the TALLRec is trained on only 64 rec-tuning samples (*i.e.*, in the 64-shot training setting). Figure (b) demonstrates the performance tendency of TALLRec's variants and conventional sequential recommendation methods *w.r.t.* the number of training samples used, ranging from 1 to 256. TALLRec has three variants: "AT" for alpaca tuning only, "RT" for rec-tuning only, and "TALLRec" for the full version.

From the figure and table, we draw the following observations: 1) Our method significantly outperforms both traditional and LLM-based methods, verifying the superiority of tuning LLM via our TALLRec framework. TALLRec successfully unlocks the knowledge and generalization capabilities of LLMs for recommendations. 2) LLM-based methods perform similarly to random guessing (AUC≈0.5). However, the LLMs trained via TALLRec achieves significant improvements. These results demonstrate a considerable gap between recommendation and language tasks, showing the importance of using recommendation data for rec-tuning on LLMs. 3) Traditional recommender methods consistently yield inferior performance under our few-shot training settings, implying that traditional methods are incapable of quickly learning the recommendation capability with limited training samples. 4) GRU-BERT and DROS-BERT do not show significant improvement over their backend models, GRU4Rec and DROS. This indicates that purely adding textual descriptions cannot enhance the traditional recommender models in the few-shot training setting.

3.2 Ablation Study (RQ2)

To demonstrate the effectiveness of alpaca tuning and rec-tuning in TALLRec, we conduct ablation studies with varying K under the K-shot training setting. Specifically, we compare the performance of TALLRec with that of two variants, "AT" and "RT", where "AT" only conducts the alpaca tuning, while "RT" solely implements rec-tuning. By varying K, we further investigate the impact of the number of training samples.

We summarize the results in Figure 3 (b), from which we have the following observations: 1) The performance of "AT" significantly declines compared to that of "RT" and TALLRec, indicating the essential effect of rec-tuning, which effectively inspires the LLM's recommendation capability 2) With limited rec-tuning samples ($\leq$ 128), TALLRec generally outperforms "RT", confirming that alpaca tuning can enhance the LLM's generalization ability on new tasks, especially when the training data in the new tasks are insufficient. As the quantity of rec-tuning samples grows, the results of TALLRec and "RT" become closer. This makes sense, as the significance of generalization abilities derived from other tasks diminishes when there is an ample amount of training data for the new tasks. 3) With the increase of rec-tuning sample number, TALLRec consistently performs better than the baselines. It is attributed to rec-tuning, which can utilize limited samples to inspire the LLM's recommendation capability.

3.3 Cross-domain Generalization Analyses (RQ3)

To further investigate the generalization ability of TALLRec, we conduct experiments on cross-domain recommendations. Specifically, we tune TALLRec with different rec-tuning samples, including 1) "TALLRec (Book)", only using the samples from the Book dataset; 2) "TALLRec (Movie)", solely using samples from the Movie dataset; and 3) "TALLRec (Both)", tuned with both the Book and Movie samples. We vary K in $\{16, 64, 258\}$ under the few-shot training setting, and evaluate the models on the testing sets of Book and Movie, respectively. The results are summarized in Figure 4, from

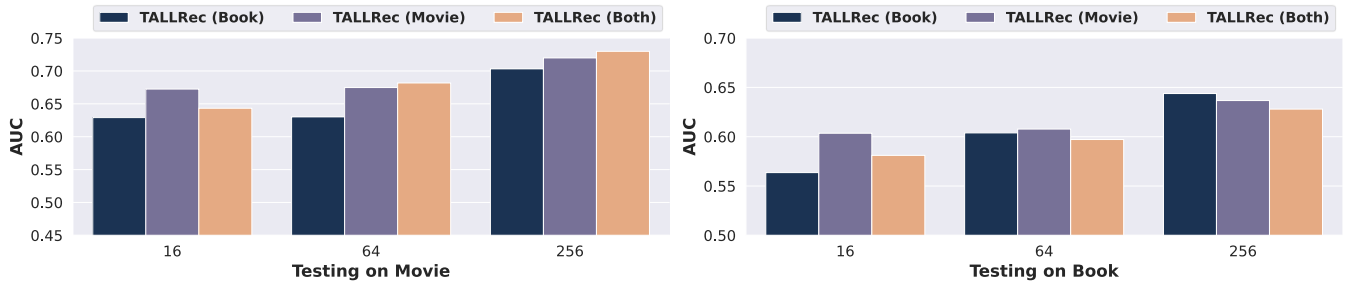

Figure 4: Cross-domain generalization performance of LRLMs trained via TALLRec using Book data (TALLRec (Book)), Movie data (TALLRec (Movie)), and both (TALLRec (Both)). The left figure shows the testing results on the Movie dataset with varying numbers of rec-tuning samples, while the right figure shows the testing results on the Book dataset.

which we can find: 1) TALLRec demonstrates remarkable cross-domain generalization ability. For instance, after tuning only on movie samples, "TALLRec (Movie)" exhibits good performance on Book data, comparable to "TALLRec (Book)". This is impressive and suggests that TALLRec has cross-domain generalization ability instead of only fitting a single domain like traditional recommenders. 2) "TALLRec (Both)" surpasses "TALLRec (Movie)" and "TALLRec (Book)" on two testing sets when the number of rec-tuning samples exceeds 64. This finding indicates that TALLRec can seamlessly integrate data from different domains to enhance its generalization performance. In future work, it is promising to pre-train TALLRec with large-scale recommendation data from heterogeneous domains.

4 RELATED WORK

• **LMs for Recommendation.** There have been several attempts to integrate language models (LMs) with recommendation systems. Despite the incorporation of LMs [14, 29], some attempts persist in utilizing traditional user/item IDs to represent users/items. Thereby, they disregard the semantic understanding capabilities of LMs, such as reviews, which other work has incorporated the language information as part of the users/items embedding [19]. In addition, other methods either utilize an undisclosed model that already possesses preliminary recommendation capabilities [6]. or employ small models to train on large-scale downstream task data [68]. Moreover, the aforementioned models are also limited to small models, while this paper is orthogonal about how to adapt large language models to recommendation tasks. In recommendation systems, there is currently little research on applying LLMs in recommendation scenarios. Those works utilize the interaction ability of GPT3.5 series models and apply In-context Learning [13, 47]. In detail, Chat-Rec [13] endeavors to harness the interaction capabilities of ChatGPT and link the ChatGPT with traditional recommendation models (e.g. MF [27], LightGCN [16]) to formulate a conversational recommendation system. NIR [47] shares a similar concept with Chat-Rec, which employs conventional recommendation models to generate candidate items, which are subjected to a three-stage multi-step prompting process for re-ranking.

• **Sequential Recommendation.** Our setup is close to the sequential recommendation, which aims to infer the user's next interaction based on users' historical interaction sequences [11, 50]. In the early time, the Markov chain plays an important role in sequential recommendation [15, 34, 40, 49]. Recently, deep learning-based methods have become mainstream. Extensive work using different kinds of neural network structures, like RNN [5, 9, 17], CNN [44, 59, 62], and attention [26, 58, 65], to model the user interaction sequences. However, limited by only using IDs to represent users and items, such work cannot fastly adapt and generalize to new scenarios. Thus, some works focus on the generalization ability of sequential recommendation models by pre-training [35, 61], data augmentation [38, 39, 51, 57], debiasing [8, 52, 67, 70], and robust optimization [55, 60]. However, they ignore the strong generalization ability of existing LLMs, leading to inadequate exploration.

5 CONCLUSION

With the advancement of LLMs, people are gradually recognizing their potential in recommendation systems [30, 63, 64]. In this work, we explored the feasibility of using LLMs for the recommendation. Our initial findings reveal that even the existing best LLM models do not perform well in recommendation tasks. To address this issue, we proposed a TALLRec framework that can efficiently align LLM with recommendation tasks through two tuning stages: alpaca tuning and rec-tuning. Our experimental results demonstrate that the LLMs trained using our TALLRec framework outperform traditional models and exhibit strong cross-domain generalization abilities. Moving forward, we plan to explore more efficient methods to activate the recommendation ability of larger models and tune LLMs to handle multiple recommendation tasks simultaneously.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (62272437), and the CCCD Key Lab of Ministry of Culture and Tourism.

REFERENCES

[1] Qingyao Ai, Ting Bai, Zhao Cao, Yi Chang, Jiawei Chen, Zhumin Chen, Zhiyong Cheng, Shoubin Dong, Zhicheng Dou, Fuli Feng, et al. 2023. Information Retrieval Meets Large Language Models: A Strategic Report from Chinese IR Community. *arXiv preprint arXiv:2307.09751* (2023).
[2] Greg Brockman, Mira Murati, Peter Welinder, and OpenAI. 2022. OpenAI API.
[3] Tom Brown, Benjamin Mann, Nick Ryder, et al. 2020. Language models are few-shot learners. *NeurIPS* 33 (2020), 1877–1901.
[4] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311* (2022).
[5] Qiang Cui, Shu Wu, Qiang Liu, Wen Zhong, and Liang Wang. 2020. MV-RNN: A Multi-View Recurrent Neural Network for Sequential Recommendation. *IEEE Trans. Knowl. Data Eng.* 32, 2 (2020), 317–331.
[6] Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. M6-Rec: Generative Pretrained Language Models are Open-Ended Recommender Systems. *arXiv preprint arXiv:2205.08084* (2022).

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*. Association for Computational Linguistics, 4171–4186.

[8] Sihao Ding, Fuli Feng, Xiangnan He, Jinqiu Jin, Wenjie Wang, Yong Liao, and Yongdong Zhang. 2022. Interpolative Distillation for Unifying Biased and Debiased Recommendation. In *SIGIR '22, July 11 - 15, 2022*. ACM, 40–49.

[9] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. 2017. Sequential user-based recurrent neural network recommendations. In *Proceedings of the eleventh ACM conference on recommender systems*. 152–160.

[10] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, et al. 2023. PaLM-E: An Embodied Multimodal Language Model. abs/2303.03378 (2023). arXiv:2303.03378

[11] Hui Fang, Danning Zhang, Yiheng Shu, and Guibing Guo. 2020. Deep Learning for Sequential Recommendation: Algorithms, Influential Factors, and Evaluations. *ACM Trans. Inf. Syst.* 39, 1 (2020), 10:1–10:42.

[12] F.Maxwell and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context.. In *ACM Transactions on Interactive Intelligent Systems (TiiS)*.

[13] Yunfan Gao et al. 2023. Chat-REC: Towards Interactive and Explainable LLMs-Augmented Recommender System. *arXiv preprint:2303.14524* (2023).

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

[15] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *2016 IEEE 16th international conference on data mining (ICDM)*. IEEE, 191–200.

[16] 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'20*. 639–648.

[17] Balázs Hidasi et al. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR*.

[18] Jordan Hoffmann et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556* (2022).

[19] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards Universal Sequence Representation Learning for Recommender Systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 585–593.

[20] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, et al. 2019. Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019*, Vol. 97. 2790–2799.

[21] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*.

[22] Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, et al. 2022. OPT-IML: Scaling Language Model Instruction Meta Learning through the Lens of Generalization. *arXiv preprint arXiv:2212.12017* (2022).

[23] Joel Jang, Seungone Kim, Seonghyeon Ye, Doyoung Kim, Lajanugen Logeswaran, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2023. Exploring the Benefits of Training Expert Language Models over Instruction Tuning. (2023). arXiv:2302.03202

[24] Vitor Jeronymo, Luiz Henrique Bonifacio, Hugo Abonizio, Marzieh Fadaee, Roberto de Alencar Lotufo, Jakub Zavrel, and Rodrigo Frassetto Nogueira. 2023. InPars-v2: Large Language Models as Efficient Dataset Generators for Information Retrieval. abs/2301.01820 (2023). arXiv:2301.01820

[25] Matthew Jin, Syed Shahriar, Michele Tufano, Xin Shi, Shuai Lu, Neel Sundaresan, and Alexey Svyatkovskiy. 2023. InferFix: End-to-End Program Repair with LLMs. (2023). arXiv:2303.07263

[26] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *ICDM*. 197–206.

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

[28] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP,7-11 November, 2021*. Association for Computational Linguistics, 3045–3059.

[29] Lei Li, Yongfeng Zhang, and Li Chen. 2023. Personalized prompt learning for explainable recommendation. *ACM Transactions on Information Systems* 41, 4 (2023), 1–26.

[30] Ruyu Li, Wenhao Deng, Yu Cheng, Zheng Yuan, Jiaqi Zhang, and Fajie Yuan. 2023. Exploring the Upper Limits of Text-Based Collaborative Filtering Using Large Language Models: Discoveries and Insights. *arXiv preprint arXiv:2305.11700* (2023).

[31] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *ACL/IJCNLP 2021, August 1-6, 2021*. Association for Computational Linguistics, 4582–4597.

[32] Percy Liang, Rishi Bommasani, Tony Lee, et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110* (2022).

[33] Xi Victoria Lin, Todor Mihaylov, et al. 2021. Few-shot learning with multilingual language models. *arXiv preprint arXiv:2112.10668* (2021).

[34] Tariq Mahmood and Francesco Ricci. 2007. Learning and adaptivity in interactive recommender systems. In *Proceedings of the ninth international conference on Electronic commerce*. 75–84.

[35] Yabo Ni, Dan Ou, Shichen Liu, et al. 2018. Perceive Your Users in Depth: Learning Universal User Representations from Multiple E-commerce Tasks. In *KDD 2018, August 19-23, 2018*. ACM, 596–605.

[36] Long Ouyang et al. 2022. Training language models to follow instructions with human feedback. *NeurIPS* 35 (2022), 27730–27744.

[37] Baolin Peng et al. 2023. Instruction Tuning with GPT-4. (2023). arXiv:2304.03277

[38] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2021. Contrastive Learning for Representation Degeneration Problem in Sequential Recommendation. (2021). arXiv:2110.05730

[39] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2022. Contrastive Learning for Representation Degeneration Problem in Sequential Recommendation. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, February 21 - 25, 2022*. ACM, 813–823.

[40] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. 811–820.

[41] Victor Sanh, Albert Webson, Colin Raffel, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv:2110.08207* (2021).

[42] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761* (2023).

[43] Dhruv Shah, Blazej Osinski, Brian Ichter, and Sergey Levine. 2022. LM-Nav: Robotic Navigation with Large Pre-Trained Models of Language, Vision, and Action. In *CoRL 2022, 14-18 December 2022 (Proceedings of Machine Learning Research, Vol. 205)*. PMLR, 492–504.

[44] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*. 565–573.

[45] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model.

[46] Hugo Touvron et al. 2023. LlaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[47] Lei Wang et al. 2023. Zero-Shot Next-Item Recommendation using Large Pretrained Language Models. *arXiv preprint arXiv:2304.03153* (2023).

[48] Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query Expansion with Large Language Models. (2023). arXiv:2303.07678

[49] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval*. 403–412.

[50] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet A. Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *IJCAI 2019, Macao, China, August 10-16, 2019*. ijcai.org, 6332–6338.

[51] Ziyang Wang, Huoyu Liu, Wei Wei, Yue Hu, Xian-Ling Mao, Shaojian He, Rui Fang, and Dangyang Chen. 2022. Multi-level Contrastive Learning Framework for Sequential Recommendation. (2022). arXiv:2208.13007

[52] Zhenlei Wang, Shiqi Shen, Zhipeng Wang, Bo Chen, Xu Chen, and Ji-Rong Wen. 2022. Unbiased Sequential Recommendation with Latent Confounders. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*. ACM, 2195–2204.

[53] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652* (2021).

[54] Jason Wei, Yi Tay, Rishi Bommasani, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682* (2022).

[55] Hongyi Wen, Xinyang Yi, Tiansheng Yao, Jiaxi Tang, Lichan Hong, and Ed H. Chi. 2022. Distributionally-robust Recommendations for Improving Worst-case User Experience. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*. ACM, 3606–3610.

[56] Ted Xiao, Harris Chan, Pierre Sermanet, Ayzaan Wahid, Anthony Brohan, Karol Hausman, Sergey Levine, and Jonathan Tompson. 2022. Robotic Skill Acquisition via Instruction Augmentation with Vision-Language Models. abs/2211.11736 (2022). arXiv:2211.11736

[57] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive Learning for Sequential Recommendation. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*. IEEE, 1259–1273.

[58] Chengfeng Xu, Jian Feng, Pengpeng Zhao, Fuzhen Zhuang, Deqing Wang, Yanchi Liu, and Victor S. Sheng. 2021. Long- and short-term self-attention network for sequential recommendation. *Neurocomputing* 423 (2021), 580–589.

[59] An Yan, Shuo Cheng, Wang-Cheng Kang, Mengting Wan, and Julian J. McAuley. 2019. CosRec: 2D Convolutional Neural Networks for Sequential Recommendation. In *CIKM 2019, November 3-7, 2019*. ACM, 2173–2176.

[60] Zhengyi Yang et al. 2023. A Generic Learning Framework for Sequential Recommendation with Distribution Shifts. (2023).

[61] Fajie Yuan, Xiangnan He, Alexandros Karatzoglou, and Liguang Zhang. 2020. Parameter-Efficient Transfer from Sequential Behaviors for User Modeling and Recommendation. In *SIGIR 2020, July 25-30, 2020*. ACM, 1469–1478.

[62] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M. Jose, and Xiangnan He. 2019. A Simple Convolutional Generative Network for Next Item Recommendation. In *WSDM 2019, February 11-15, 2019*. ACM, 582–590.

[63] 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 *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 2639–2649. https://doi.org/10.1145/3539618.3591932

[64] Jizhi Zhang, Keqin Bao, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys '23)*. Association for Computing Machinery.

[65] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, et al. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, August 10-16, 2019*. ijcai.org, 4320–4326.

[66] Yang Zhang et al. 2023. Reformulating CTR Prediction: Learning Invariant Feature Interactions for Recommendation. (2023).

[67] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. 2021. Causal Intervention for Leveraging Popularity Bias in Recommendation. In *SIGIR '21, July 11-15, 2021*. ACM, 11–20.

[68] Zizhuo Zhang and Bang Wang. 2023. Prompt Learning for News Recommendation. *arXiv preprint arXiv:2304.05263* (2023).

[69] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A Survey of Large Language Models. *CoRR* abs/2303.18223 (2023). https://doi.org/10.48550/arXiv.2303.18223 arXiv:2303.18223

[70] Yu Zheng et al. 2021. Disentangling User Interest and Conformity for Recommendation with Causal Embedding. In *WWW '21*. 2980–2991.

[71] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. 2005. Improving Recommendation Lists through Topic Diversification. In *Proceedings of the 14th International Conference on World Wide Web (WWW '05)*. Association for Computing Machinery, 22–32.