RDRec: Rationale Distillation for LLM-based Recommendation

Xinfeng Wang[†], **Jin Cui**[†], **Yoshimi Suzuki**[‡], and **Fumiyo Fukumoto**[‡]
[†]Graduate School of Engineering
[‡]Interdisciplinary Graduate School
University of Yamanashi, Kofu, Japan
{g22dtsa7, g22dtsa5, ysuzuki, fukumoto}@yamanashi.ac.jp

Abstract

Large language model (LLM)-based recommender models that bridge users and items through textual prompts for effective semantic reasoning have gained considerable attention. However, few methods consider the underlying rationales behind interactions, such as user preferences and item attributes, limiting the reasoning capability of LLMs for recommendations. This paper proposes a rationale distillation recommender (RDRec), a compact model designed to learn rationales generated by a larger language model (LM). By leveraging rationales from reviews related to users and items, RDRec remarkably specifies their profiles for recommendations. Experiments show that RDRec achieves state-of-the-art (SOTA) performance in both top-N and sequential recommendations. Our source code is released at https://github.com/WangXFng/RDRec.

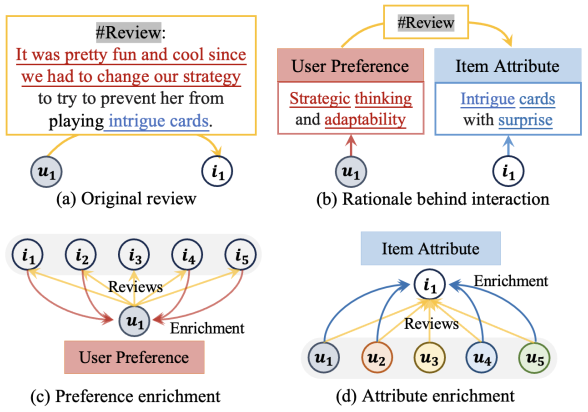
#Review:
It was pretty fun and cool since we had to change our strategy to try to prevent her from playing intrigue cards.
u_1
i_1
(a) Original review
#Review
User Preference
Strategic thinking and adaptability
Item Attribute
Intrigue cards with surprise
u_1
i_1
(b) Rationale behind interaction
i_1 i_2 i_3 i_4 i_5
Reviews
u_1
Enrichment
User Preference
(c) Preference enrichment
Item Attribute
i_1
Enrichment
Reviews
u_1 u_2 u_3 u_4 u_5
(d) Attribute enrichment

Figure 1: Illustration of our motivation. (a) denotes the review after a purchase and (b) refers to the rationale of the purchase distilled by LLMs. (c) and (d) indicate the preference and attribute enrichment, respectively.

1 Introduction

Large language models (LLMs) with powerful reasoning capabilities have been extensively studied for recommendations, including news and item recommendations (Li et al., 2022; Wei et al., 2023; Huang et al., 2023), explainable recommendations (Yang et al., 2023; Cheng et al., 2023), and zero-/few-shot and cold-start recommendations (He et al., 2023; Sanner et al., 2023). Several attempts have leveraged knowledge of LLMs to improve recommendation performance, such as enhancing embedding initialization (Harte et al., 2023), reranking candidates (Yue et al., 2023), and learning representation (Ren et al., 2023; Lin et al., 2023; Lei et al., 2023; Viswanathan et al., 2023). A straightforward approach is to integrate user and item IDs into LMs through prompt learning (Liu et al., 2023), including discrete prompts to find alternative words to represent IDs, continuous prompts to directly feed ID vectors into a pre-trained model (Sun et al., 2019), and hybrid prompts (Li et al., 2023a; Zhang and Wang, 2023). Recently, Geng et al. (2022) present a P5 paradigm to transform user–item interactions, user sequential behaviors, and reviews into text-to-text prompts for LLMs. This enables P5 to capture deeper semantics for LLM-based recommendations. Li et al. (2023b) enhance P5 by a prompt distillation, resulting in significant improvement and reductions in inference time.

However, they pay no attention to mining the rationale behind each interaction, such as user preferences and item attributes, which hampers the reasoning capabilities of LLMs. As an example, in Fig. 1 (a), a user review for an item says: "*It was pretty fun and cool since we had to change our strategy (user preference) to try to prevent her from playing intrigue cards (item attributes).*" The user prefers strategic thinking in the game, and intrigue cards symbolize item characteristics. This introduces noise into the user's profile, as the user leans towards a strategic game rather than merely cards. This suggests that the original review without intermediate prompts prevents the model from learning to understand the rationale behind the interaction.

The Chain-of-Thought (CoT) prompting (Wei et al., 2022; Wang et al., 2023a) that promises

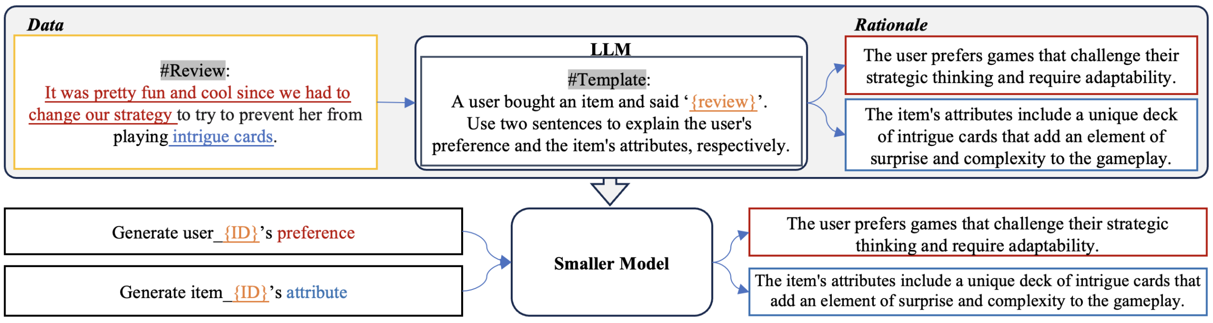
Data
#Review:
It was pretty fun and cool since we had to change our strategy to try to prevent her from playing intrigue cards.
LLM
#Template:
A user bought an item and said '{review}'. Use two sentences to explain the user's preference and the item's attributes, respectively.
Rationale
The user prefers games that challenge their strategic thinking and require adaptability.
The item's attributes include a unique deck of intrigue cards that add an element of surprise and complexity to the gameplay.
Generate user_{ID}'s preference
Generate item_{ID}'s attribute
Smaller Model
The user prefers games that challenge their strategic thinking and require adaptability.
The item's attributes include a unique deck of intrigue cards that add an element of surprise and complexity to the gameplay.

Figure 2: Illustration of rationale distillation with LLMs via the chain-of-thought (CoT) prompting.

LLMs to decompose intermediate rationales, has been widely applied for rationale extraction (Wang et al., 2023b; Zhang et al., 2023b; McKee et al., 2023; Zhu et al., 2023). More recently, Hsieh et al. (2023) utilize the CoT prompting to distill rationales via LLMs to train smaller models. Inspired by this, we propose a compact recommender model to learn the interaction rationales, i.e., user preferences and item attributes, distilled from reviews using a larger LM. In this way, the model acquires clear textual knowledge with less noise (e.g., "intrigue cards" that may hinder understanding the user's preference for "strategic games" in Fig. 1 (b)). This enables the model to derive more specified user and item profiles from all reviews given by the user or regarding the item for recommendations, as illustrated in Fig. 1 (cd).

The main contributions of this paper can be summarized as follows. (1) We propose a compact RDRec model that effectively specifies user and item profiles by distilling interaction rationales from relevant reviews using a larger LM, and (2) RDRec consistently outperforms SOTA baselines on three real-world datasets in both sequential and top-N recommendations.

2 RDRec Framework

We present an RDRec model consisting of two stages, an interaction rationale distillation and a rationale-aware recommendation.

2.1 Interaction Rationale Distillation

Inspired by the recent works (Hsieh et al., 2023; Miao et al., 2023) that employ LLMs to produce training data for smaller models, we distill user preferences and item attributes from reviews by using the following prompt template: "*A user bought an item and said '{review}'. Use two sentences to explain the user's preference and the item's attributes, respectively.*" As illustrated in Fig. 2, a review feeds into LLMs with the prompt template. The output is user preferences and item attributes.

Formally, given a user–item interaction triplet $(u, i, r_{u,i})$ where u, i, and $r_{u,i}$ indicate a user, an item, and a review, respectively, we generate a quadruplet $(u, i, p_{u,i}, a_{u,i})$ through rationale distillation. Here, $p_{u,i}$ and $a_{u,i}$ refer to the distilled user preference and item attribute, respectively.

2.2 Rationale-aware Recommendation

The RDRec uses PrOmpt Distillation (POD) (Li et al., 2023b) as its backbone. POD converts three recommendation tasks into LLM-based text generation tasks, and then distills continuous prompt vectors from task templates. These tasks are (i) sequential recommendations, predicting the next item through the user's ordered interactions, (ii) top-N recommendations, recommending the top N items not yet engaged with by the user, and (iii) explanation generation for a user's interactions.

In contrast to POD, we incorporate an additional rationale generation task, consisting of a user preference generation and an item attribute generation. Specifically, following POD, we first distill prompt vectors ("<P4>" and "<P5>" in Fig. 3) from the templates of "*Generate user_{#u}'s preference*" and "*Generate item_{#i}'s attribute*", where $\#u$ and $\#i$ denote the user and item IDs. Then, we concatenate prompt vectors with user and item IDs as the input, and the generated preference $p_{u,i}$ and attribute $a_{u,i}$ as the output to train the model. To address the token composing issue (i.e., the token of "user_123" is often tokenized by LLMs as a sequence of ["user", "_", "12" and "3"]), we use the whole-word embedding (Geng et al., 2022) to treat each sequence of ID tokens as a complete unit, making it distinguishable as a word.

Fig. 3 illustrates the input and output example of the four tasks. We define a pair of input-output

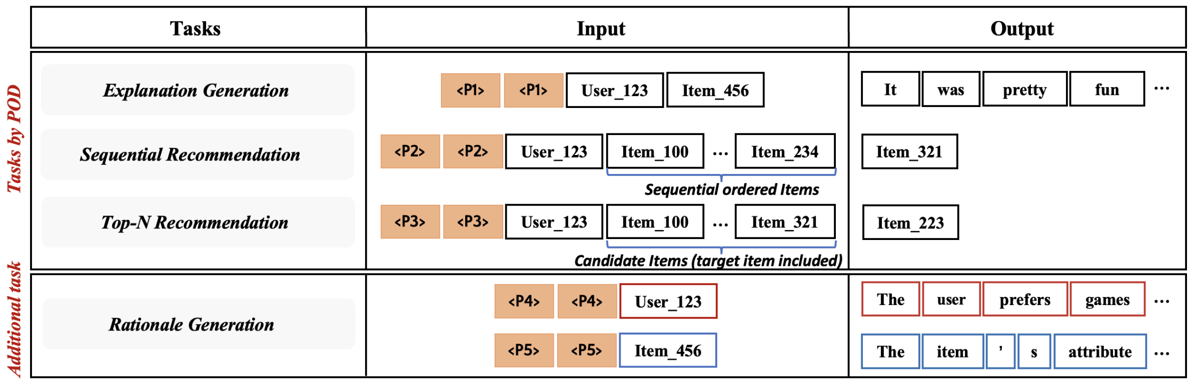

Figure 3: Illustration of input and output of four tasks by RDRec in the prompt distillation setting.

words as $X = [x_1, ..., x_{|X|}]$ and $Y = [y_1, ..., y_{|Y|}]$, respectively. We then concatenate the tokens of the input with prompt vectors and obtain $[\mathbf{x}_1, ..., \mathbf{x}_{|X|}, \mathbf{p}_1, ..., \mathbf{p}_{|P|}]$. After adding the whole-word representation $[\mathbf{w}_1, ..., \mathbf{w}_{|X|+|P|}]$, we feed them into the smaller model in RDRec to obtain a probability distribution $p(y|Y_{<t}, X)$ over a vocabulary at each step t, where $Y_{<t}$ denotes the tokens generated before step t. We adopt a log-likelihood loss function to optimize the model parameters Θ:

$$\mathcal{L}_\Theta = \frac{1}{|\mathcal{D}|} \sum_{(X,Y)\in\mathcal{D}} \frac{1}{|Y|} \sum_{t=1}^{|Y|} -\log p(y|Y_{<t}, X), \tag{1}$$

where $\mathcal{D}$ denotes the training set consisting of all input-output pairs for four tasks. $|\mathcal{D}|$ and $|Y|$ denote the amount of training samples and the number of tokens in the output sequence, respectively.

2.3 Model Optimization and Inference

Following POD, we shuffle the input-output pairs of four tasks and randomly select samples from each task in a specified proportion. We thereafter mixed these samples to train the RDRec model. During inference, we employ a beam search algorithm to generate results by selecting the word with the highest likelihood from the vocabulary.

3 Experiment

3.1 Experimental Setup

Datasets and Metrics. Consistent with POD, we performed experiments on three public datasets, i.e., Sports & Outdoors, Beauty, and Toys & Games, which are collected from the Amazon dataset[1]. Each record in the dataset contains a user ID, an item ID, a rating, a textual review, and a timestamp. We split each dataset into training, validation, and test sets with a ratio of 8:1:1. The statistics of datasets are provided in Table 1. To evaluate the recommendation performance, we utilized the evaluation metrics of hit rate (HR)@k (H@k) and normalized discounted cumulative gain (NDCG)@k (N@k) with $k \in \{1, 5, 10\}$.

[1]https://www.amazon.com/

Dataset	#User	#Item	#Review	Avg.	Density (%)
Sports	48,993	34,298	296,337	8.3	0.0453
Beauty	22,363	12,101	198,502	8.9	0.0734
Toys	19,804	22,086	167,597	8.6	0.0724

Table 1: Statistics of dataset. "#User", "#Item", "#Review", and "Avg." denote the number of users, items, reviews, and average user reviews, respectively.

Baselines. We compared RDRec with ten baselines for sequential recommendations: CASER (Tang and Wang, 2018), HGN (Ma et al., 2019), GRU4Rec (Hidasi et al., 2015), BERT4Rec (Sun et al., 2019), FDSA (Zhang et al., 2019), SASRec (Kang and McAuley, 2018), S^3-Rec (Zhou et al., 2020), P5 (Geng et al., 2022), RLS (Chu et al., 2023) and POD (Li et al., 2023b). We compared RDRec with five baselines for top-N recommendations: MF (Koren et al., 2009), MLP (Cheng et al., 2016), P5 (Geng et al., 2022), RLS (Chu et al., 2023) and POD (Li et al., 2023b).

Implementation. For a fair comparison, RDRec used T5-small (Raffel et al., 2020) as the smaller model, aligning with the baselines P5 and POD. We used Llama-2-7b (Touvron et al., 2023) as the larger LM. We reported a 10-trial T-test to show the robustness of RDRec. Our RDRec was implemented and experimented with Pytorch on Nvidia GeForce RTX 3090 (24GB memory). The Appendix A.1 provides further details.

Models	Sports				Beauty				Toys			
	H@5	N@5	H@10	N@10	H@5	N@5	H@10	N@10	H@5	N@5	H@10	N@10
Caser	0.0116	0.0072	0.0194	0.0097	0.0205	0.0131	0.0347	0.0176	0.0166	0.0107	0.0270	0.0141
HGN	0.0189	0.0120	0.0313	0.0159	0.0325	0.0206	0.0512	0.0266	0.0321	0.0221	0.0497	0.0277
GRU4Rec	0.0129	0.0086	0.0204	0.0110	0.0164	0.0099	0.0283	0.0137	0.0097	0.0059	0.0176	0.0084
BERT4Rec	0.0115	0.0075	0.0191	0.0099	0.0203	0.0124	0.0347	0.0170	0.0116	0.0071	0.0203	0.0099
FDSA	0.0182	0.0122	0.0288	0.0156	0.0267	0.0163	0.0407	0.0208	0.0228	0.0140	0.0381	0.0189
SASRec	0.0233	0.0154	0.0350	0.0192	0.0387	0.0249	0.0605	0.0318	0.0463	0.0306	0.0675	0.0374
S^3-Rec	0.0251	0.0161	0.0385	0.0204	0.0387	0.0244	0.0647	0.0327	0.0443	0.0294	0.0700	0.0376
P5	0.0387	0.0312	0.0460	0.0336	0.0508	0.0379	0.0664	0.0429	0.0648	0.0567	0.0709	0.0587
RSL	0.0392	0.0330	0.0512	0.0375	0.0508	0.0381	0.0667	0.0446	0.0676	0.0583	0.0712	0.0596
POD	0.0497	0.0399	0.0579	0.0422	0.0559	0.0420	0.0696	0.0471	0.0692	0.0589	0.0749	0.0601
Ours	**0.0505**	**0.0408**	**0.0596**	**0.0433**	**0.0601**	**0.0461**	**0.0743**	**0.0504**	**0.0723**	**0.0593**	**0.0802**	**0.0605**
Impv (%).	**1.6**	**2.2**	**2.8**	**2.5**	**7.5***	**9.8***	**6.7***	**7.1***	**4.4***	**0.6**	**7.1***	**0.7**
p-value	6.3e-1	5.1e-1	2.7e-1	3.8e-1	8.1e-3	2.4e-3	2.1e-2	2.5e-2	1e-2	5.8e-1	1.7e-5	5.9e-1

Table 2: Performance comparison on sequential recommendation. **Bold**: Best, underline: Second best. "*" indicates that the improvement is statistically significant (p-value < 0.05) in the 10-trial T-test. All of the baselines are reported by the papers (Geng et al., 2022; Chu et al., 2023; Li et al., 2023b), except for the POD model.

Models	Sports					Beauty					Toys				
	H@1	H@5	N@5	H@10	N@10	H@1	H@5	N@5	H@10	N@10	H@1	H@5	N@5	H@10	N@10
MF	0.0314	0.1404	0.0848	0.2563	0.1220	0.0311	0.1426	0.0857	0.2573	0.1224	0.0233	0.1066	0.0641	0.2003	0.0940
MLP	0.0351	0.1520	0.0927	0.2671	0.1296	0.0317	0.1392	0.0848	0.2542	0.1215	0.0252	0.1142	0.0688	0.2077	0.0988
P5	0.0726	0.1955	0.1355	0.2802	0.1627	0.0608	0.1564	0.1096	0.2300	0.1332	0.0451	0.1322	0.0889	0.2023	0.1114
RSL	0.0892	0.2092	0.1502	0.3001	0.1703	0.0607	0.1612	0.1110	0.2209	0.1302	0.0389	0.1423	0.0825	0.1926	0.1028
POD	0.0927	0.2105	0.1539	0.2889	0.1782	0.0846	0.1931	0.1404	0.2677	0.1639	0.0579	0.1461	0.1029	0.2119	0.1244
Ours	**0.1285**	**0.2747**	**0.2033**	**0.3683**	**0.2326**	**0.1203**	**0.2572**	**0.1902**	**0.3380**	**0.2160**	**0.0660**	**0.1655**	**0.1171**	**0.2375**	**0.1398**
Impv. (%)	**38.6***	**30.5***	**32.1***	**27.5***	**30.5***	**42.2***	**33.2***	**35.8***	**26.3***	**31.8***	**13.9***	**13.2***	**13.8***	**12.1***	**12.4***
p-value	2.3e-14	1.1e-14	2.8e-15	1.1e-16	5.0e-15	3.8e-15	2.0e-15	1.7e-15	2.7e-15	2.1e-15	5.6e-7	4.4e-8	2.4e-8	1.2e-8	9.8e-9

Table 3: Comparison on top-N recommendation. The T-test shows the results by RDRec and the second-best, POD.

3.2 Experimental Results

Tables 2 and 3 show comparative results between RDRec and baselines. We can see that the RDRec consistently surpasses the runner-ups, POD and RSL, with the improvement of 0.5 $\sim$ 9.8% in H@k and N@k for sequential recommendations, and 12.1 $\sim$ 42.2% in H@k and N@k for top-N recommendations, where $k \in \{1, 5, 10\}$. This highlights the effectiveness of learning interaction rationales to improve both recommendation tasks.

We also observed that RDRec exhibits greater improvement in top-N recommendations compared to sequential recommendations. This indicates that specifying user preferences and item attributes is more beneficial to recommending top-N unknown candidates, whereas sequential recommenders rely more on capturing correct behavioral patterns for predicting the user's next choice.

We conducted an ablation experiment to examine the rationale distillation. The result in Table. 4 shows that distilling user preferences and item attributes from reviews is advantageous for both sequential and top-N recommendations. We can see that specifying item profiles is generally more effective for top-N recommendation, whereas specifying user profiles is more effective for sequential recommendation on the Sports and Beauty datasets.

UsP	ItA	Sports		Beauty		Toys	
		H@10	N@10	H@10	N@10	H@10	N@10
Sequential recommendation							
✗	✗	0.0566	0.0408	0.0705	0.0479	0.0768	0.0573
✓	✗	0.0581	0.0425	0.0729	0.0494	0.0787	0.0589
✗	✓	0.0573	0.0411	0.0712	0.0492	0.0788	0.0593
✓	✓	**0.0596**	**0.0433**	**0.0743**	**0.0504**	**0.0802**	**0.0605**
Top-N recommendation							
✗	✗	0.2977	0.1850	0.2777	0.1701	0.2200	0.1284
✓	✗	0.3509	0.2200	0.3080	0.1912	0.2214	0.1307
✗	✓	0.3513	0.2249	0.3275	0.2048	0.2321	0.1370
✓	✓	**0.3683**	**0.2326**	**0.3380**	**0.2160**	**0.2375**	**0.1398**

Table 4: Ablation study. "w/o X" denotes the removed parts. "UsP" and "ItA" indicate the distillation of user preferences and item attributes, respectively.

3.3 Error Analysis of Sequential Recommendation

We conducted an error analysis to examine the sequential recommendations by RDRec. We identified two noteworthy error cases:

Case (i). RDRec may prioritize the next item based on a user's earlier interactions rather than recent ones. One reason is that the Transformer (Vaswani et al., 2017) in T5 excels in capturing long-term dependencies, while it may cause RDRec to pay less attention to recent interactions. This suggests to enhance its self-attention (Fan et al., 2022) or develop short-term prompt-aware templates for LLM-based sequential recommendations.

Ratio	Sports		Beauty		Toys	
EG:RG:SR:TR	H@10	N@10	H@10	N@10	H@10	N@10
Sequential recommendation						
1 : 1 : 1 : 1	**0.0596**	**0.0433**	**0.0743**	**0.0504**	0.0789	0.0594
1 : 1 : 2 : 1	0.0593	0.0431	0.0735	0.0502	0.0790	0.0601
1 : 1 : 1 : 3	0.0592	0.0426	0.0702	0.0445	**0.0802**	**0.0605**
Top-N recommendation						
1 : 1 : 1 : 1	0.3261	0.2022	0.2855	0.1854	0.2214	0.1307
1 : 1 : 2 : 1	0.2822	0.1722	0.2693	0.1584	0.1872	0.1037
1 : 1 : 1 : 3	**0.3683**	**0.2160**	**0.3380**	**0.2160**	**0.2375**	**0.1398**

Table 5: Performance on the sample ratios of various tasks. "EG", "RG", "SR" and "TR" denote explanation and rationale generation, and sequential and top-N recommendations, respectively.

Case (ii). RDRec often disregards popular items for users because they do not align with their sequential patterns. One possible reason for this is that, during training, RDRec selects random subsequences from the user interaction sequence and predicts the last item of each subsequence. This process emphasizes sequential patterns but possibly sacrifices the model's capability to identify popular items. This suggests introducing a popularity-based interaction graph to help the model be aware of popular high-order neighbors.

3.4 In-Depth Analysis of RDRec

To better understand the RDRec, we conducted in-depth experiments and analysis. The Appendix A.2 provides further analyses.

Effect of sample ratios. We observe from Table 5 that on the Toys dataset, increasing the ratio of top-N samples for training RDRec improves sequential recommendations, while in major cases a higher ratio of sequential samples always harms top-N recommendations. One reason is that the training strategy of sequential tasks prioritizes sequential patterns while compromising its ability to detect unknown items.

Computational complexity. Both Llama2 and T5 are Transformer-based models, with computational complexity of $\mathcal{O}(L^2)$, where L is the number of word tokens. Consequently, RDRec's computational complexity relies on user interaction count rather than the number of users and items. Compared with other complex ID-based methods, such as graph convolution network-based approaches with $\mathcal{O}((M+N)^2)$ (He et al., 2020; Yu et al., 2022; Wang et al., 2023c, 2024), where M and N are the numbers of users and items, respectively, and $(M{+}N) \gg L$ in Table 1, RDRec exhibits reduced computational demands, thereby rendering it suitable for deployment in large-scale applications.

Study of rationale distillation. We investigated the rationale distillation and obtained two findings. One is that, even when a user negatively reviews an item, the LLM objectively specifies user requirements and item attributes. For instance, in the following input, the customer advises not buying the book unless the kids are truly interested in it. However, many others provide positive comments, such as "*The toy was really nice.*" and "*Fun little toy to match the book*."

Input:
My Nephew is all about trucks and machines it's cute for him but unless the kid's really into the book or just general construction I wouldn't bother.

This indicates that objective profiles (e.g., a book and its content) are more crucial than users' subjective opinions in real-world recommendations. We found that the generated item attributes by the LLM are relatively objective which is shown as follows:

Output:
The user prefers items that are cute and appealing to children, but not necessarily related to construction or machines.

The item's attributes include being a colorful and engaging picture book that teaches children about different construction vehicles.

This could be a reason for the noticeable improvement in performance by learning rationales.

The other observation is that, when a review is extremely short, the prompt could urge the LLM to produce hallucinations during rationale distillations. Recently, Zhang et al. (2023a) have proposed to mitigate hallucinations of LLM-based recommender to enhance its performance. This is a rich space for further exploration (Liu et al., 2022; Gao et al., 2023; Peng et al., 2023).

4 Conclusion

We proposed a compact RDRec model to learn the underlying rationales for interactions generated by a larger LM. By learning rationales from all related reviews, RDRec effectively specifies user and item profiles for recommendations. Experimental results showed the effectiveness of our RDRec. Future work involves (i) exploring better prompts for sequential recommendations, and (ii) enhancing explanation generation in RDRec.

Acknowledgements

We would like to thank anonymous reviewers for their thorough comments and suggestions. This work is supported by the China Scholarship Council (No.202208330093) and JKA (No.2023M-401).

Ethics Statement

This paper does not involve the presentation of a new dataset, an NLP application, and the utilization of demographic or identity characteristics information.

Limitation

The hallucination issue during rationale distillation remains unsolved. Additionally, RDRec faces an unfaithful reasoning problem, misinterpreting user opinions about candidate items despite delivering correct recommendations.

References

Hao Cheng, Shuo Wang, Wensheng Lu, Wei Zhang, Mingyang Zhou, Kezhong Lu, and Hao Liao. 2023. Explainable recommendation with personalized review retrieval and aspect learning. In *the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 51–64.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 7–10.

Zhixuan Chu, Hongyan Hao, Xin Ouyang, Simeng Wang, Yan Wang, Yue Shen, Jinjie Gu, Qing Cui, Longfei Li, Siqiao Xue, et al. 2023. Leveraging large language models for pre-trained recommender systems. *arXiv preprint arXiv:2308.10837*.

Ziwei Fan, Zhiwei Liu, Yu Wang, Alice Wang, Zahra Nazari, Lei Zheng, Hao Peng, and Philip S Yu. 2022. Sequential recommendation via stochastic self-attention. In *Proceedings of the ACM Web Conference 2022*, pages 2036–2047.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, et al. 2023. Rarr: Researching and revising what language models say, using language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16477–16508.

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

Jesse Harte, Wouter Zorgdrager, Panos Louridas, Asterios Katsifodimos, Dietmar Jannach, and Marios Fragkoulis. 2023. Leveraging large language models for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1096–1102.

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

Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. 2023. Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pages 720–730.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*.

Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. 2023. Recommender ai agent: Integrating large language models for interactive recommendations. *arXiv preprint arXiv:2308.16505*.

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

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

Yuxuan Lei, Jianxun Lian, Jing Yao, Xu Huang, Defu Lian, and Xing Xie. 2023. Recexplainer: Aligning large language models for recommendation model interpretability. *arXiv preprint arXiv:2311.10947*.

Jian Li, Jieming Zhu, Qiwei Bi, Guohao Cai, Lifeng Shang, Zhenhua Dong, Xin Jiang, and Qun Liu. 2022. Miner: multi-interest matching network for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 343–352.

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

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

Xinyu Lin, Wenjie Wang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2023. A multi-facet paradigm to bridge large language model and recommendation. *arXiv preprint arXiv:2310.06491*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Tianyu Liu, Yizhe Zhang, Chris Brockett, Yi Mao, Zhifang Sui, Weizhu Chen, and Bill Dolan. 2022. A token-level reference-free hallucination detection benchmark for free-form text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Chen Ma, Peng Kang, and Xue Liu. 2019. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 825–833.

Daniel McKee, Justin Salamon, Josef Sivic, and Bryan Russell. 2023. Language-guided music recommendation for video via prompt analogies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14784–14793.

Zhongjian Miao, Wen Zhang, Jinsong Su, Xiang Li, Jian Luan, Yidong Chen, Bin Wang, and Min Zhang. 2023. Exploring all-in-one knowledge distillation framework for neural machine translation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2929–2940.

Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

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*.

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

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

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

Hugo Touvron, 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*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Vijay Viswanathan, Luyu Gao, Tongshuang Wu, Pengfei Liu, and Graham Neubig. 2023. Datafinder: Scientific dataset recommendation from natural language descriptions. In *the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10288–10303.

Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2023a. Towards understanding chain-of-thought prompting: An empirical study of what matters. In *the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2717–2739.

Peifeng Wang, Zhengyang Wang, Zheng Li, Yifan Gao, Bing Yin, and Xiang Ren. 2023b. Reasoning implicit sentiment with chain-of-thought prompting. In *the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1171–1182.

Xinfeng Wang, Fumiyo Fukumoto, Jin Cui, Yoshimi Suzuki, Jiyi Li, and Dongjin Yu. 2023c. Eedn: Enhanced encoder-decoder network with local and global context learning for poi recommendation. In

Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 383–392.

Xinfeng Wang, Fumiyo Fukumoto, Jin Cui, Yoshimi Suzuki, and Dongjin Yu. 2024. Nfarec: A negative feedback-aware recommender model. *arXiv preprint arXiv:2404.06900*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2023. Llmrec: Large language models with graph augmentation for recommendation. *arXiv preprint arXiv:2311.00423*.

Zhengyi Yang, Jiancan Wu, Yanchen Luo, Jizhi Zhang, Yancheng Yuan, An Zhang, Xiang Wang, and Xiangnan He. 2023. Large language model can interpret latent space of sequential recommender. *arXiv preprint arXiv:2310.20487*.

Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 1294–1303.

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

An Zhang, Leheng Sheng, Yuxin Chen, Hao Li, Yang Deng, Xiang Wang, and Tat-Seng Chua. 2023a. On generative agents in recommendation. *arXiv preprint arXiv:2310.10108*.

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

Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, Xiaofang Zhou, et al. 2019. Feature-level deeper self-attention network for sequential recommendation. In *IJCAI*, pages 4320–4326.

Zizhuo Zhang and Bang Wang. 2023. Prompt learning for news recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 227–237.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1893–1902.

Yingjie Zhu, Jiasheng Si, Yibo Zhao, Haiyang Zhu, Deyu Zhou, and Yulan He. 2023. Explain, edit, generate: Rationale-sensitive counterfactual data augmentation for multi-hop fact verification. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13377–13392.

A Appendix

A.1 Experimental Details

This section provides further experimental results and implementation setup. We focus on the fourth task as Geng et al. (2022) have conducted a thorough study for the others.

A.1.1 Effect of Various Sample Ratios

Table 6 shows the effect of various sample ratios on the recommendation performance. We can see that the sample ratio of various tasks for pretraining the model would influence the recommendation performance. Specifically, on the Toys dataset, increasing the ratio of top-N samples sometimes improves both sequential and top-N recommendations. In contrast, a higher ratio of sequential samples often negatively affects the performance of top-N recommendations across all datasets. The reason is that during training, RDRec selects random user interaction subsequences and predicts the last item of each subsequence. This process emphasizes sequential patterns, although possibly sacrifices the model's capability to identify popular items.

A.1.2 Execution Time

Table 7 shows the execution time in various stages by RDRec on three datasets. These results were obtained through Nvidia GeForce RTX 3090 (24GB memory). We can see that RDRec efficiently makes inferences for recommendations with a small backbone, while the interaction rationale distillation and pre-training are time-consuming. Fortunately, these processes are only required once.

A.1.3 Implementation Details

For a fair comparison, all the hyperparameters of RDRec are in the same setting as POD. Specifically, both the encoder and decoder consist of 6 layers with each layer comprising an 8-headed attention layer. The vocabulary of T5 contains a total number of 32,100 tokens, with an embedding dimensionality of 512. We iteratively and randomly sampled a segment from a user's item sequence for training the sequential recommendation task. The number of negative items for top-N recommendation is set to 99 for both training and evaluation. We used the AdamW optimizer (Loshchilov and Hutter, 2017). We set the number of prompt vectors to 3 for all tasks, the batch size for training all three tasks to 64, and the learning rate to 0.001 for the Sports dataset and 0.0005 for both the Beauty and Toys datasets. We exploit the discrete prompt templates for different tasks from (Geng et al., 2022). During training, we save a checkpoint if the total validation loss of the model in all tasks is the lowest for the current epoch. If this doesn't occur 5 times, we terminate the training process and load the best checkpoint for evaluation. At the inference stage, we set the number of beams at 20 for sequential and top-N recommendations. For generation tasks, we apply group beam search with the number of beams and beam groups set to 21 and 3, respectively.

A.1.4 Baselines

To evaluate the performance of sequential and top-N recommendations, we compared our RDRec with twelve baselines:

- **MF** (Koren et al., 2009) accesses the inner product between user and item latent factors for predicting users' preference for candidates.
- **GRU4Rec** (Hidasi et al., 2015) regards the entire item sequence of each user as the user's session to recommend.
- **MLP** (Cheng et al., 2016) exploits a stack of nonlinear layers to learn user and item embeddings for making recommendations.
- **CASER** (Tang and Wang, 2018) treats user interactions as images and employs 2-dimensional convolutions to capture sequential patterns.
- **SASRec** (Kang and McAuley, 2018) exploits Markov Chains to excavate short-term semantics in users' sequential patterns.
- **HGN** (Ma et al., 2019) exploits a novel gating strategy to model users' long- and short-term interests in candidate items.
- **BERT4Rec** (Sun et al., 2019) proposes to leverage the BERT-style cloze task for the sequential recommender algorithm.
- **FDSA** (Zhang et al., 2019) incorporates item features with item sequences of users to perform recommendations.
- **S^3-Rec** (Zhou et al., 2020) learns users' latent behavioral features via employing a self-supervised learning paradigm.
- **P5** (Geng et al., 2022) converts three different recommendation tasks into textual generation tasks using LLMs for recommendations.
- **RSL** (Chu et al., 2023) adopts novel training and inference strategies to deliver LLM-based recommendations.
- **POD** (Li et al., 2023b) refines P5 through prompt distillation to make efficient and precise recommendations.

Ratio	Sports				Beauty				Toys			
EG:RG:SR:TR	H@5	N@5	H@10	N@10	H@5	N@5	H@10	N@10	H@5	N@5	H@10	N@10
Sequential recommendation												
1 : 1 : 1 : 1	**0.0503**	**0.0402**	**0.0596**	**0.0433**	**0.0601**	**0.0461**	**0.0743**	**0.0504**	0.0716	0.0579	0.0789	0.0594
1 : 1 : 2 : 1	0.0501	0.0398	0.0593	0.0431	0.0595	0.0457	0.0735	0.0502	0.0713	0.0581	0.0790	0.0601
1 : 1 : 3 : 1	0.0496	0.0399	0.0578	0.0420	0.0573	0.0417	0.0662	0.0452	0.0713	0.0583	0.0792	0.0601
1 : 1 : 1 : 2	0.0489	0.0374	0.0571	0.0387	0.0565	0.0419	0.0715	0.0466	0.0717	0.0588	0.0799	0.0602
1 : 1 : 1 : 3	0.0483	0.0369	0.0592	0.0426	0.0547	0.0395	0.0702	0.0445	**0.0723**	**0.0593**	**0.0802**	**0.0605**
Top-N recommendation												
1 : 1 : 1 : 1	0.2381	0.1750	0.3261	0.2022	0.2136	0.1516	0.2885	0.1854	0.1482	0.1062	0.2144	0.1307
1 : 1 : 2 : 1	0.2042	0.1476	0.2822	0.1722	0.1845	0.1350	0.2693	0.1584	0.1253	0.0876	0.1872	0.1037
1 : 1 : 3 : 1	0.1524	0.1080	0.2101	0.1298	0.1424	0.1024	0.2178	0.1359	0.1118	0.0780	0.1803	0.0998
1 : 1 : 1 : 2	0.2439	0.1810	0.3303	0.2067	0.2372	0.1784	0.3237	0.2030	0.1579	0.1091	0.2221	0.1339
1 : 1 : 1 : 3	**0.2747**	**0.2033**	**0.3683**	**0.2326**	**0.2572**	**0.1902**	**0.3380**	**0.2160**	**0.1655**	**0.1171**	**0.2375**	**0.1398**

Table 6: Performance comparison on various sample ratios for training RDRec. "EG", "RG", "SR" and "TR" denote explanation generation, rationale generation, and sequential and top-N recommendations, respectively.

Datasets	**Stages**			
	Distillation	**Pre-training**	**SR**	**TR**
Sports	16h46m28s	16h23m23s	15m03s	18m23s
Beauty	11h50m14s	12h45m12s	13m33s	16m07s
Toys	09h13m05s	08h39m37s	16m25s	18m21s

Table 7: Execution time in various stages. "SR" and "TR" represent the cumulative inference time for all users in sequential and top-N recommendations, respectively. "h", "m", and "s" refer to "hours" and "minutes", and "seconds" respectively.

A.2 Further Analyses

A.2.1 Hallucination by LLMs

We observed that when a review is too short, the LLM might produce hallucinations. The following example illustrates a rationale with hallucinations generated by the LLM:

Input:
This is a fantastic game.

Output:
The user prefers games with engaging storylines .

The item's attributes include a realistic game world , immersive sound effects , and smooth gameplay .

The contents of "*engaging storylines*", "*immersive sound effects*" and "*smooth gameplay*" marked by gray are hallucinations overly inferred by the LLM. Toward this, mitigating hallucinations of LLM-based recommender is a rich space for future exploration (Liu et al., 2022; Gao et al., 2023; Peng et al., 2023; Zhang et al., 2023a).

A.2.2 Effect of Explanation Generation

We observed that RDRec can generate correct explanations in many cases, such as the explanation "*This is a great product for the price*," for the provided review "*very good quality for the price*."

However, RDRec sometimes recommends candidates correctly but provides explanations that completely differ from the user's review. For instance, the generated explanation is, "*Absolutely great product*," whereas the user's actual review is, "*I wouldn't recommend this for painting your full nail*." One possible reason is that RDRec has learned to prioritize predicting user-item interaction over considering the rationale for making recommendations. This is a challenging yet intriguing path to further improve RDRec.

A.2.3 The Whole-Word Embedding

To address the token composing issue (i.e., the token of "user_1234" is often tokenized by the tokenizer of LLMs as a sequence of ["user", "_", "12" and "34"]), we employed the whole-word embedding (Geng et al., 2022) to ensure that each sequence of ID tokens is a complete unit and can be distinguished from a word.

It is noteworthy that the whole-word embedding will not cause scalability issues because we only need to identify which tokens represent the same user (or item). For instance, given a token list ["P1", "P2", "P3", "user", "_", "12", "34", "item", "_", "98", "76"], the index list over the whole-word embedding vocabulary is [0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2]. Since the number of negative samples is set to 99 and the average user interaction is less than 9 in our datasets, an embedding matrix (512 * 512) with a maximum incremental number of 512 is sufficient. Even if a user's interaction count exceeds 512, we only need to expand the whole-word embedding matrix, which is acceptable for a real-world deployment.