Uncovering ChatGPT's Capabilities in Recommender Systems

Sunhao Dai*
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
sunhaodai@ruc.edu.cn

Ninglu Shao*
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
ninglu_shao@ruc.edu.cn

Haiyuan Zhao*
School of Information, Renmin University of China
Beijing, China
haiyuanzhao@ruc.edu.cn

Weijie Yu
School of Information Technology and Management, University of International Business and Economics
Beijing, China
yuweijie23@gmail.com

Zihua Si
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
zihua_si@ruc.edu.cn

Chen Xu
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
xc_chen@ruc.edu.cn

Zhongxiang Sun
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
sunzhongxiang@ruc.edu.cn

Xiao Zhang
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
zhangx89@ruc.edu.cn

Jun Xu†
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
junxu@ruc.edu.cn

ABSTRACT

The debut of ChatGPT has recently attracted significant attention from the natural language processing (NLP) community and beyond. Existing studies have demonstrated that ChatGPT shows significant improvement in a range of downstream NLP tasks, but the capabilities and limitations of ChatGPT in terms of recommendations remain unclear. In this study, we aim to enhance ChatGPT's recommendation capabilities by aligning it with traditional information retrieval (IR) ranking capabilities, including point-wise, pair-wise, and list-wise ranking. To achieve this goal, we re-formulate the aforementioned three recommendation policies into prompt formats tailored specifically to the domain at hand. Through extensive experiments on four datasets from different domains, we analyze the distinctions among the three recommendation policies. Our findings indicate that ChatGPT achieves an optimal balance between cost and performance when equipped with list-wise ranking. This research sheds light on a promising direction for aligning ChatGPT with recommendation tasks. To facilitate further explorations in this area, the full code and detailed original results are open-sourced at https://github.com/rainym00d/LLM4RS.

CCS CONCEPTS

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

KEYWORDS

ChatGPT, large language model, recommender systems

ACM Reference Format:
Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering ChatGPT's Capabilities in Recommender Systems. 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/10.1145/3604915.3610646

1 INTRODUCTION

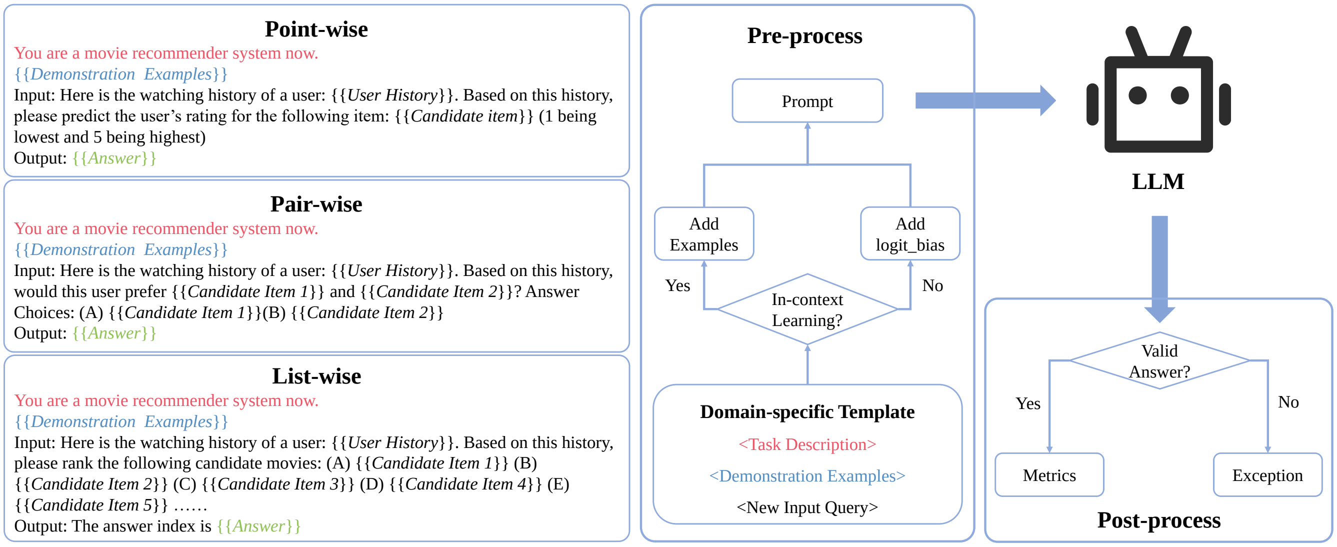
Large language models (LLMs), such as ChatGPT developed by OpenAI [23], have recently gained significant attention from the natural language processing (NLP) community and beyond. These LLMs possess a versatile nature and extensive world knowledge, allowing them to be applied not only in various NLP tasks [3, 5, 16], but also in domains such as education [11, 19, 22], medicine [1, 2, 29], search [20, 26, 37] and law [4, 8, 38].

Meanwhile, previous research has indicated that off-the-shelf pre-trained language models (LMs) can be directly used as recommenders by adapting recommendation tasks into multi-token cloze tasks using prompts [25, 34, 43]. Hence, a natural research question arises regarding how to effectively align LLMs with recommendation capabilities and whether LLMs can work as few-shot or zero-shot recommender systems.

The primary objective of recommender systems is to alleviate information overload by providing personalized top-K item ranking lists for users [32]. In information retrieval (IR), previous studies

[*] The first three authors contributed equally.
[†] Jun Xu is the corresponding author. Work partially done at Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education.

RecSys '23, September 18–22, 2023, Singapore, Singapore
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0241-9/23/09.
https://doi.org/10.1145/3604915.3610646



Point-wise

You are a movie recommender system now.
{{*Demonstration Examples*}}
Input: Here is the watching history of a user: {{*User History*}}. Based on this history, please predict the user's rating for the following item: {{*Candidate item*}} (1 being lowest and 5 being highest)
Output: {{*Answer*}}

Pair-wise

You are a movie recommender system now.
{{*Demonstration Examples*}}
Input: Here is the watching history of a user: {{*User History*}}. Based on this history, would this user prefer {{*Candidate Item 1*}} and {{*Candidate Item 2*}}? Answer Choices: (A) {{*Candidate Item 1*}}(B) {{*Candidate Item 2*}}
Output: {{*Answer*}}

List-wise

You are a movie recommender system now.
{{*Demonstration Examples*}}
Input: Here is the watching history of a user: {{*User History*}}. Based on this history, please rank the following candidate movies: (A) {{*Candidate Item 1*}} (B) {{*Candidate Item 2*}} (C) {{*Candidate Item 3*}} (D) {{*Candidate Item 4*}} (E) {{*Candidate Item 5*}} ……
Output: The answer index is {{*Answer*}}

Figure 1: The overall evaluation framework of LLMs for recommendation. The left part demonstrates examples of how prompts are constructed to elicit each of the three ranking capabilities. The right part outlines the process of employing LLMs to perform different ranking tasks and conduct evaluations.

have commonly utilized three approaches to construct these ranking lists: point-wise, pair-wise, and list-wise [18, 42]. Consequently, in this paper, our specific focus is on probing the recommendation capabilities of LLMs by aligning them with these three ranking perspectives. Detailed formulation of these three ranking perspectives could be seen in Section 3.

To investigate the potential of LLMs in recommendation tasks from these three ranking perspectives, we begin by reformulating the three capabilities into prompts that are tailored to the specific domain and serve as input for LLMs. We then conduct an empirical analysis of ChatGPT and other popular LLMs from OpenAI on four widely-used recommendation benchmarks from different knowledge-rich domains. To the best of our knowledge, this is the first empirical study to probe the capabilities of ChatGPT in recommender systems from different ranking perspectives.

Major Findings. In summary, we have the following major findings after empirical experiments:

- ChatGPT shows consistent advantages in all three ranking capabilities compared with other LLMs.
- ChatGPT is good at list-wise and pair-wise ranking while less good at point-wise ranking.
- ChatGPT can outperform traditional recommendation models with limited training data.
- Considering the improvements with cost, we recommend list-wise ranking for LLM-based recommenders in practice.
- ChatGPT exhibits potential in explainable recommendations and a good understanding of item similarity.

We hope that this preliminary evaluation of ChatGPT in recommendation can provide new perspectives on both assessing the capabilities of LLMs and utilizing LLMs, such as ChatGPT, to enhance recommender systems.

2 BACKGROUND

2.1 Large Language Models

Pioneering studies [3, 27] demonstrated that LLMs can perform a diverse range of tasks without requiring gradient updates, solely based on textual instructions or a few examples. This has drawn significant attention towards improving the capabilities of LLMs. Previous studies [15] have investigated the performance limits of pre-trained language models (PLMs) by training larger models, as they have noted that augmenting the model or data size typically enhances the model's ability on downstream tasks, such as Megatron-turing NLG [35] with 530B parameters, Gopher [28] with 280B parameters, Ernie 3.0 Titan [39] with 260B parameters, BLOOM [33] with 175B parameters, and PaLM [5] with 540B parameters. These LLMs have exhibited exceptional performance on challenging tasks, showcasing new abilities that were not apparent in smaller pre-trained language models. For a more comprehensive overview of LLMs, we would recommend referring to [44].

2.2 Language Models for Recommendation

The remarkable success of pre-trained LMs in NLP community has motivated researchers in recommender systems to explore their potential in recommendation tasks. Existing works can be categorized into two types: (i) utilizing LMs training strategies to reformulate and model recommendation tasks, such as BERT4Rec (*masked language modeling*) [36], UnisRec (*pre-train and finetune*)[13], P5 (*pre-train and prompting*) [9] and (ii) using LMs to obtain better representations of users and items as extra features based on textual information [40]. More recently, some researchers have explored leveraging off-the-shelf pre-trained LMs as recommender systems by reformulating the recommendation tasks with prompts as multi-token cloze tasks [25, 34, 43]. In this paper, we aim to conduct a preliminary evaluation of ChatGPT's potential and limitations in recommender systems.

3 PROBING CHATGPT FOR RECOMMENDATION CAPABILITIES

In this section, we leverage prompts to adapt point-wise, pair-wise, and list-wise ranking tasks, enabling off-the-shelf LLMs to effectively tackle these tasks.

3.1 Three Ranking Capabilities in Recommender Systems

The core objective of personalized recommendation is to rank candidate items based on user preferences. To accomplish this, current learning-to-rank (LTR) methods empower different capabilities to recommender systems via corresponding loss functions, including point-wise ranking capability, pair-wise ranking capability and list-wise ranking capability [18]. Formally, given a user $u \in \mathcal{U}$ and k candidate items $\{i_1, i_2, \cdots, i_k\} \subset \mathcal{I}$, each user-item pair's representation is encoded as $\mathbf{x}_{u,i}$. The above three capabilities can be formulated as follows:

Definition 1. *(point-wise ranking capability) The recommender system learns to predict the preference score of each item i for each user u via a point-wise scoring function $\Phi_{\text{point}}(\cdot)$: $s(i \mid u) = \Phi_{\text{point}}(\mathbf{x}_{u,i})$ The preference score s is then used to rank the items for each user. The common used loss function in point-wise ranking includes mean squared error (MSE) [31] and binary cross entropy (BCE) [12].*

Definition 2. *(pair-wise ranking capability) The recommender system learns to compare pairs of items i_m and i_n for each user u and predict which item is preferred via a pair-wise scoring function $\Phi_{\text{pair}}(\cdot)$: $s(i_m \succ i_n \mid u) = \Phi_{\text{pair}}(\mathbf{x}_{u,i_m}, \mathbf{x}_{u,i_n})$ The system then ranks the items based on the relative preference score s. The pairwise hinge loss [14] or Bayesian personalized ranking loss (BPR) [30] are the typical loss functions utilized in pair-wise ranking.*

Definition 3. *(List-wise ranking capability) The recommender system learns to directly predict the preference score of a list of items $\{i_1, i_2, \cdots, i_k\}$ for each user u via a list-wise scoring function $\Phi_{\text{list}}(\cdot)$: $s(i_1 \mid u), s(i_2 \mid u), \cdots, s(i_k \mid u) = \Phi_{\text{list}}(\mathbf{x}_{u,i_1}, \mathbf{x}_{u,i_2}, \cdots, \mathbf{x}_{u,i_k})$ The system then sorts the items based on the predicted scores. The list-wise loss, e.g., sampled softmax loss [7] is employed to optimize the recommendation model.*

3.2 Reformulate and Adapt Recommendation with Prompts

To obtain above capabilities of recommendation, current recommendation models employ corresponding loss functions for supervised learning. However, the supervised learning schema often fails in data sparsity scenarios (e.g., cold start problems [10] and long-tailed items [24]). In contrast, LLMs have a stronger generalization capability in these data sparsity scenarios and achieve promising performances in few-shot and even zero-shot tasks. In this empirical study, we assume that LLMs already have the above three capabilities, and all we need to do is to trigger these capabilities through prompt tuning. To this end, we adopt the recent successful practice of in-context learning [3] and instruction tuning [6], and we express the aforementioned three capabilities as three tasks with domain-specific prompts.

Figure 1 illustrates how we employ prompt tuning to elicit three ranking capabilities from LLMs. As shown in Figure 1 (left), our prompt consists of three components: (i) *Task description* I refers to the process of enabling the LLM to comprehend the particular domain in which it is required to perform recommendation tasks. The task description is designed to be domain-aware, which enhances LLM's perception of pertinent knowledge. (ii) *Demonstration examples* $\mathcal{D} = \{f(\mathbf{h}_1, \mathbf{c}_1, \mathbf{y}_1), \cdots, f(\mathbf{h}_N, \mathbf{c}_N, \mathbf{y}_N)\}$ (i.e., N-shot in-context learning), where $\mathbf{h}$ denotes the historical interacted items of a user, $\mathbf{c}$ denotes the candidate items which need to be ranked, $\mathbf{y}$ denotes the predictions given by LLMs, and $f(\cdot)$ is the function for transforming the examples into designed prompt templates. The demonstration examples facilitate the LLM's comprehension of the current task. (iii) *New input query* of a given user $f(\mathbf{h}', \mathbf{c}' \mid u)$, which needs to be answered by LLMs. For three ranking tasks, the corresponding candidate items $\mathbf{c}$ are constructed as follows:

$$\mathbf{c} = \begin{cases} \{i\} & \text{for point-wise ranking capability,} \\ \{i_m, i_n\} & \text{for pair-wise ranking capability,} \\ \{i_1, i_2, \cdots, i_k\} & \text{for list-wise ranking capability.} \end{cases}$$

As depicted in Figure 1 (right), LLMs will utilize the different ranking capabilities elicited through different prompts to make predictions $\hat{y}'$:

$$\begin{aligned} \hat{y}'_i &= LLM_{\text{point}}\left(I, \mathcal{D}, f(\mathbf{h}', \mathbf{c}' \mid u)\right), \\ \hat{y}'_{i_m \succ i_n} &= LLM_{\text{pair}}\left(I, \mathcal{D}, f(\mathbf{h}', \mathbf{c}' \mid u)\right), \\ \hat{y}'_{i_1}, \hat{y}'_{i_2}, \cdots, \hat{y}'_{i_k} &= LLM_{\text{list}}\left(I, \mathcal{D}, f(\mathbf{h}', \mathbf{c}' \mid u)\right). \end{aligned}$$

Then the output answer will be checked manually, and the valid answers will be utilized for further evaluation, while the invalid answers will be excluded. For more details about the prompts, please refer to the link[1].

4 EXPERIMENTS

In this section, we conduct experiments to evaluate ChatGPT and GPT-3.5s to answer the following research questions:

RQ1: How do these LLMs perform on different ranking capabilities across various recommendation domains?

RQ2: How do the LLMs-based recommenders compare with traditional collaborative filtering methods?

RQ3: How much cost of these LLMs-based recommenders on different ranking capabilities?

RQ4: How does the number of prompt shots affect the performance of LLMs-based recommenders?

4.1 Experimental Settings

4.1.1 Datasets. To better probe the different capabilities of ChatGPT and GPT-3.5s on personalized recommendation, we conducted evaluations on datasets from four different domains.

Movie: We use the widely-adopted MovieLens-1M[2] dataset that contains 1M user ratings for movies.

[1] https://github.com/rainym00d/LLM4RS/blob/main/assets/prompts.pdf
[2] https://grouplens.org/datasets/movielens/1m/

Table 1: Overall performance of different models on four datasets from different domains. Bold indicates the best result for each row and '_' indicates the best result for each LLM. 'random' denotes recommendation based on a random policy. 'pop' denotes recommendation based on items' popularity judged by the number of interactions.

Domain	Metric	random	pop	text-davinci-002			text-davinci-003			gpt-3.5-turbo (ChatGPT)		
				point-wise	pair-wise	list-wise	point-wise	pair-wise	list-wise	point-wise	pair-wise	list-wise
Movie	Compliance Rate	-	-	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	99.98%	100.00%
	NDCG@3	0.4262	0.4761	0.5416	0.5728	0.4990	0.4618	0.5441	0.5564	**0.5912**	0.5827	0.5785
	MRR@3	0.3667	0.4103	0.4824	0.5071	0.4363	0.3998	0.4763	0.4950	**0.5260**	0.5162	0.5167
Book	Compliance Rate	-	-	99.96%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	99.98%	99.80%
	NDCG@3	0.4262	0.4999	0.4889	0.5298	0.4290	0.4585	0.5293	0.4597	0.5075	0.5350	**0.5395**
	MRR@3	0.3667	0.4340	0.4247	0.4646	0.3690	0.3993	0.4665	0.4040	0.4495	0.4774	**0.4800**
Music	Compliance Rate	-	-	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	99.96%	99.80%
	NDCG@3	0.4262	0.4094	0.4623	0.4681	0.4277	0.4732	0.5072	0.4506	0.5201	0.5439	**0.5567**
	MRR@3	0.3667	0.3470	0.4030	0.4082	0.3750	0.4113	0.4448	0.4000	0.4605	0.4830	**0.4950**
News	Compliance Rate	-	-	99.80%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	99.60%
	NDCG@3	0.4262	**0.5444**	0.4483	0.4550	0.5059	0.4880	0.4892	0.4742	0.4826	0.4991	0.5094
	MRR@3	0.3667	**0.4840**	0.3879	0.3936	0.4497	0.4271	0.4294	0.4173	0.4246	0.4354	0.4515

Table 2: Rank of different capabilities of different LLMs-based recommendation models on four datasets from different domains.

Domain	text-davinci-002	text-davinci-003	gpt-3.5-turbo (ChatGPT)
Movie	pair-wise > point-wise ≫ list-wise	list-wise ≈ pair-wise ≫ point-wise	point-wise > pair-wise ≈ list-wise
Book	pair-wise ≫ point-wise ≫ list-wise	pair-wise ≫ list-wise ≈ point-wise	list-wise > pair-wise ≫ point-wise
Music	pair-wise > point-wise ≫ list-wise	pair-wise ≫ point-wise ≫ list-wise	list-wise > pair-wise ≫ point-wise
News	list-wise ≫ pair-wise ≈ point-wise	pair-wise ≈ point-wise > list-wise	list-wise > pair-wise > point-wise

Book: We use the "Books" subset of Amazon[3] dataset that contains user ratings for books.

Music: We use the "CDs & Vinyl" subset of Amazon[3] to conduct experiments on the music domain.

News: We use the MIND-small[4] dataset as the benchmark for news domain.

Following the common practices [12, 21, 41], for the Movie, Book, and Music datasets, we treat ratings above 3 as positive feedbacks (labeled as 1) and otherwise as negative feedbacks (labeled as 0). For the News dataset, we used the original binary feedback labels. In the experiments, we use the titles of the items as description in the prompt.

4.1.2 Evaluation Protocols. After processing, we random sample 500 records on each dataset for evaluation due to the expensive cost. For all experiments, we follow the existing practice [34] and pair one positive item with four randomly sampled negative items as the candidate item list. We set the number of shots as 1 for pair-wise and list-wise, and 2 for point-wise. We report top-K Normalized Discounted Cumulative Gain (NDCG@K) and Mean Reciprocal Rank (MRR@K) with $K = 3$. Furthermore, considering that LLMs may generate some illegal output, that is, results that are not in the candidate set, we introduce the metric "Compliance Rate" to compare the compliance rates between different models, which is defined as the proportion of the number of valid results generated to all test samples, i.e., $\frac{\text{Number of Valid Answers}}{\text{Number of Test Samples}}$.

4.2 RQ1: Overall Performance

Table 1 shows the results of different LLMs on four different domains. We have the following observation and conclusions:

ChatGPT and GPT3.5s performed much better than the random recommendation in almost all cases. Specifically, all three LLMs achieve significant improvements than the random recommendation policy on four domains, e.g., average improvements with 24.71% on the point-wise task in terms of $NDCG@3$ on the Movie Dataset. Additionally, most answers of LLMs are compliant due to the capability of in-context learning. These results reveal that LLMs have the potential to facilitate recommender systems.

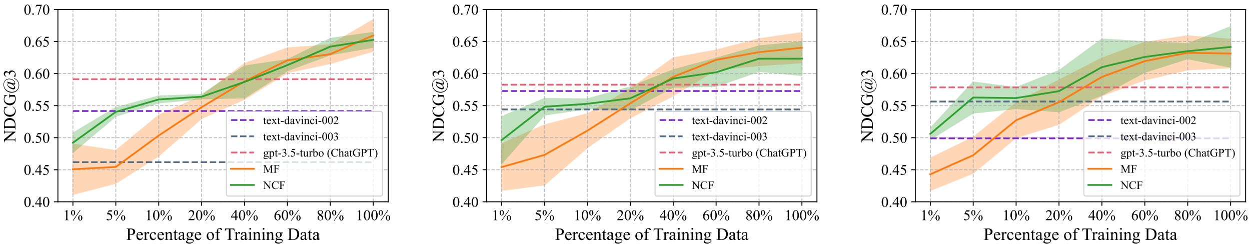
In comparison to the text-davinci-002 and text-davinci-003, ChatGPT exhibits better performance on almost all evaluation metrics for all three ranking capabilities. For instance, ChatGPT outperformed the other LLMs in 22 out of 24 comparisons, including two ranking metrics, three ranking capabilities, and four domain datasets. The only two exceptions were for point-wise ranking in the news domain when compared to text-davinci-003. We attribute ChatGPT's strong performance to its exceptional capacity for language understanding and reasoning, which allows it to effectively comprehend item similarity and make informed ranking decisions.

ChatGPT performs better with list-wise ranking except in the movie domain. On the other hand, text-davinci-002 and text-davinci-003 perform better with pair-wise ranking in most cases. To provide a clear comparison, we have summarized the ranking of the three LLMs with different ranking capabilities in

[3] http://jmcauley.ucsd.edu/data/amazon/
[4] https://msnews.github.io/

Table 3: Performance of different LLMs with zero-shot and few-shot examples on Movie dataset. Bold indicates the best result for each row and '_' indicates the best result for each wise of each LLM.

Model	Metric	random	pop	point-wise		pair-wise		list-wise	
				zero-shot	few-shot	zero-shot	few-shot	zero-shot	few-shot
text-davinci-002	NDCG@3	0.4264	0.4761	0.5168	0.5416	0.5253	**0.5728**	0.4544	0.4990
	MRR@3	0.3667	0.4103	0.4519	0.4824	0.4643	**0.5071**	0.3950	0.4363
text-davinci-003	NDCG@3	0.4264	0.4761	0.4674	0.4618	0.5249	**0.5441**	0.5062	0.5564
	MRR@3	0.3667	0.4103	0.4092	0.3998	0.4633	**0.4763**	0.4450	0.4950
gpt-3.5-turbo (ChatGPT)	NDCG@3	0.4264	0.4761	0.5413	**0.5912**	0.5833	0.5827	N/A	0.5785
	MRR@3	0.3667	0.4103	0.4742	**0.5260**	0.5243	0.5162		0.5167



(a) point-wise

(b) pair-wise

(c) list-wise

Figure 2: Comparison with collaborative filtering models in terms of different percentages of training data on Movie dataset. The shaded area indicates the 95% confidence intervals of t-distribution under 5 different experiments with random seeds.

Table 2. Note that pair-wise ranking tends to be better than point-wise ranking in most cases (11 out of 12), although it requires more inference cost due to the need for pair-wise comparisons. We will delve deeper into the cost analysis in **RQ3**.

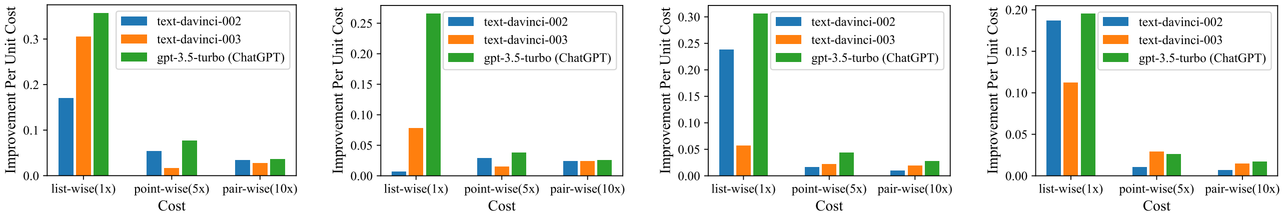
All LLMs-based recommenders outperform the popularity recommendantion policy in recommending movies, books, and music, but they underperform in the news domain. This phenomenon could be explained by the fact that news recommendation relies more on popularity, while other domains are more personalized. The speed of news delivery is another possible factor. Due to the time-sensitive and rapidly changing nature of news recommendation, there is often insufficient interaction data available for each news in the LLMs training corpus. Conversely, in the other three domains, the item descriptions and interaction data are more abundant, making LLMs works better on them. Overall, this observation suggests that while off-the-shelf LLMs-based recommenders can be effective in many domains, they may not be suitable for some domain and may require further exploration.

We also conduct experiments using zero-shot prompts (i.e., without examples). However, with the original zero-shot prompt, we find more than 50% of cases were invalid and challenging to evaluate. To address this, we utilize logit_bias[5] to control the output tokens. Due to the page limitation, we provide the detailed results in the link[6]. Overall, the results highlight the potential of LLMs as recommendation systems, as they outperform random and popularity-based policies in the zero-shot setting. Furthermore, as expected, LLMs under few-shot settings outperform those under zero-shot settings in most cases, demonstrating the effectiveness of few-shot prompts in-context learning.

4.3 RQ2: Comparison with Collaborative Filtering Models

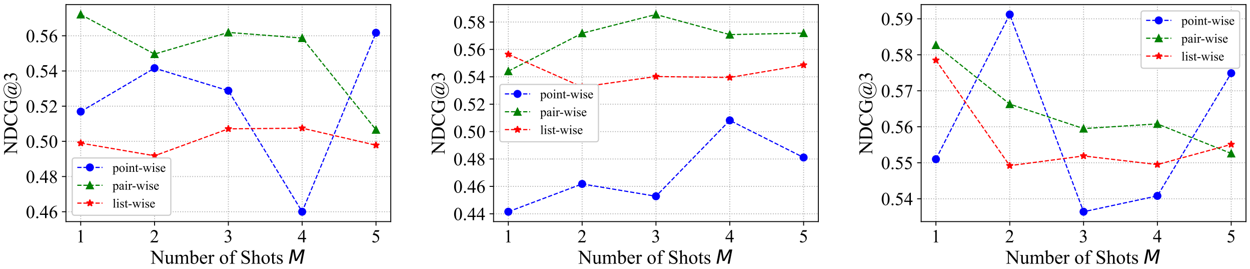
Given that the LLMs used in the previous experiments were not trained on recommendation data, we investigate the amount of training data required for traditional recommendation models to achieve performance comparable to or better than LLMs. Specifically, we chose the most representative traditional recommendation models, Matrix Factorization (MF) [17] as well as Neural Collaborative Filtering (NCF) [12], and evaluated their performance after training on varying proportions of data. For a fair comparison, we carefully tune the parameters of MF and NCF. We then compared their performance to that of LLMs. All experiments are conducted 5 times on the Movie dataset, and the averaged results and their 95% confidence intervals of *NDCG*@3 were illustrated in Figure 2. As expected, the performance of MF and NCF improves with increasing amounts of training data. Also, we can observe that off-the-shelf LLM-based models outperform MF and NCF when there are only a few training data available, i.e., less than 40% for ChatGPT with all three ranking capabilities. Note that LLM-based recommendation

[5] https://platform.openai.com/docs/api-reference/completions/create#completions/create-logit_bias
[6] https://github.com/rainym00d/LLM4RS/blob/main/assets/Supplementary_Material.pdf



(a) Movie (b) Book (c) Music (d) News

Figure 3: Improvement of *NDCG*@3 per unit cost and five shots examples on four datasets. '1x 5x 10x' denote the cost of list-wise, point-wise, and pair-wise, respectively.



(a) text-davinci-002 (b) text-davinci-003 (c) gpt-3.5-turbo (ChatGPT)

Figure 4: Impact of the number of shots prompts in LLMs on Movie dataset.

models do not require training data but rather a few samples in the prompt to help understand the recommendation task. Therefore, we conclude that LLM-based recommendation models can be applied in practice to mitigate the cold start problem.

4.4 RQ3: Performance Scaling by Cost

Although the LLMs have better performance on pair-wise or list-wise ranking as presented in Table 1, we need to consider the costs associated with these performance improvements. Specifically, we calculate the improvement per unit cost for each LLM: $\frac{\frac{V_{LLM}-V_{random}}{V_{random}}}{cost_{LLM}}$, where V_{LLM} denotes the metric value of the LLM, V_{random} denotes the metric value of random recommendation, $cost_{LLM}$ denotes the cost of ranking one user's candidate item list. Referring to Figure 1 (left), let us define the $cost_{LLM}$. For list-wise ranking, only one prompt input is needed to obtain LLM's ranking for all candidate items. For point-wise ranking, N prompt inputs are required to obtain LLM's ranking for all candidate items (where N is the number of candidate items). For pair-wise ranking, $\frac{N(N-1)}{2}$ prompt inputs are required to obtain all ranking results. In our experimental settings, N is set to 5. The costs of point-wise, pair-wise, and list-wise ranking are denoted as **5x**, **10x** and **1x**, respectively. Figure 3 demonstrates the improvement per unit cost of each LLMs. It can be found that almost all three LLMs has the best improvement per unit cost in list-wise ranking, except text-davinci-002 on the Book dataset. Moreover, point-wise ranking and pair-wise ranking have similar improvement per unit cost. Although pair-wise ranking may achieve better performance than point-wise ranking in absolute metrics, the requirement to run multiple prompts for pair-wise ranking results in additional cost. Overall, we recommend to conduct list-wise ranking for recommendation tasks in practice, due to its decent performance and lower cost.

4.5 RQ4: Performance Under Different Shots Examples

Previous studies in NLP have emphasized that the number of examples M is important for in-context learning. To assess the impact of M in LLMs for recommendation, we conducted experiments on Movie dataset by varying M from 1 to 5. Figure 4 illustrates the performances of different M in terms of *NDCG*@3 of ChatGPT and GPT3.5s. Surprisingly, we observe that the best results did not always correspond to the maximum number of examples. One possible explanation is that while more example shots can provide more context and information for LLMs to understand the recommendation task, they may also introduce more noise, causing LLMs to learn unhelpful patterns. Therefore, the optimal number of prompt shots may depend on the specific LLM, task, and dataset.

4.6 Case Study of Exceptions

It is worth noting that the LLM may generate some invalid answers even under few-shot in-context learning, leading to a compliance rate may be less than 100%, as shown in Table 1. For instance, Table 4 highlights two exceptional cases of answers from ChatGPT with pair-wise ranking, where both cases lack a correct answer because

Table 4: Case Study of Exceptions. The green is the answer of ChatGPT.

Case 1	Case 2
You are a movie recommender system now. {{*Examples*}} Input: Here is the watching history of a user: Aliens, E.T. the Extra-Terrestrial, Contact, The Matrix, X-Men. Based on this history, would this user prefer The Fox and the Hound or Steamboat Willie? Answer Choices: (A) The Fox and the Hound (B) Steamboat Willie Output: The answer index is N/A as neither option is relevant to the user's watching history.	You are a book recommender system now. {{*Examples*}} Input: Here is the reading history of a user: The Cellist of Sarajevo, After I'm Gone: A Novel, The Reason I Jump: The Inner Voice of a Thirteen-Year-Old Boy with Autism, The Serpent of Venice: A Novel, We Are All Completely Beside Ourselves: A Novel. Based on this history, would this user prefer The Secret Life of Bees or The Help? Answer Choices: (A) The Secret Life of Bees (B) The Help Output: The answer index is N/A. It is difficult to determine the user's preference based on this reading history as neither book is similar in genre or theme to the books they have read.

they are the pair of two negative samples. Surprisingly, ChatGPT does not simply respond with 'A' or 'B' as seen in the in-context learning examples. Instead, it recognizes that these two items are unrelated and not similar to the user history interactions. For example, in case 1, the user watching histories are all science fiction movies but the answer choices are all cartoons. These responses demonstrate that ChatGPT can understand how to recommend based on the user interaction histories and what is the similarity between items. However, limited by our existing evaluation methods, these answers are considered non-compliant. Therefore, we suggest exploring additional perspectives for evaluating LLMs as recommenders beyond learning to rank, as LLMs have the potential to play a larger role in explainable recommendation.

5 CONCLUSION

In this paper, we conduct a preliminary evaluation for probing off-the-shelf LLMs for recommendation from the point-wise, pair-wise, and list-wise perspectives. Specifically, we reformulate the above ranking capabilities into corresponding domain-aware prompts and evaluate the performance of ChatGPT in each ranking capability on different domains. The results on four datasets indicate the superiority of ChatGPT in recommendations among all three ranking capabilities. We also observe that LLMs excel at list-wise and pair-wise ranking, but are not proficient in point-wise ranking in most cases. Furthermore, ChatGPT shows the potential for mitigating the cold start problem and explainable recommendation.

ACKNOWLEDGMENTS

This work was funded by the National Key R&D Program of China (2019YFE0198200), Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, Intelligent Social Governance Interdisciplinary Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. Supported by fund for building world-class universities (disciplines) of Renmin University of China.

REFERENCES

[1] Fares Antaki, Samir Touma, Daniel Milad, Jonathan El-Khoury, and Renaud Duval. 2023. Evaluating the performance of chatgpt in ophthalmology: An analysis of its successes and shortcomings. *medRxiv* (2023), 2023–01.

[2] James RA Benoit. 2023. ChatGPT for Clinical Vignette Generation, Revision, and Evaluation. *medRxiv* (2023), 2023–02.

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

[4] Jonathan H Choi, Kristin E Hickman, Amy Monahan, and Daniel Schwarcz. 2023. Chatgpt goes to law school. *Available at SSRN* (2023).

[5] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311* (2022).

[6] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416* (2022).

[7] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, Shilad Sen, Werner Geyer, Jill Freyne, and Pablo Castells (Eds.). ACM, 191–198. https://doi.org/10.1145/2959100.2959190

[8] Jiaxi Cui, Zongjian Li, Yang Yan, Bohua Chen, and Li Yuan. 2023. ChatLaw: Open-Source Legal Large Language Model with Integrated External Knowledge Bases. *arXiv preprint arXiv:2306.16092* (2023).

[9] 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.

[10] Jyotirmoy Gope and Sanjay Kumar Jain. 2017. A survey on solving cold start problem in recommender systems. In *2017 International Conference on Computing, Communication and Automation (ICCCA)*. IEEE, 133–138.

[11] Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How Close is ChatGPT to Human Experts? Comparison Corpus, Evaluation, and Detection. *arXiv preprint arXiv:2301.07597* (2023).

[12] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.

[13] 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.

[14] Thorsten Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Edmonton, Alberta, Canada) *(KDD '02)*. Association for Computing Machinery, New York, NY, USA, 133–142. https://doi.org/10.1145/775047.775067

[15] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361* (2020).

[16] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916* (2022).

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

[18] Tie-Yan Liu et al. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 3, 3 (2009), 225–331.

[19] Muneer M Alshater. 2022. Exploring the role of artificial intelligence in enhancing academic performance: A case study of ChatGPT. *Available at SSRN* (2022).

[20] Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023. Zero-Shot Listwise Document Reranking with a Large Language Model. *arXiv preprint arXiv:2305.02156* (2023).

[21] Kelong Mao, Jieming Zhu, Jinpeng Wang, Quanyu Dai, Zhenhua Dong, Xi Xiao, and Xiuqiang He. 2021. SimpleX: A simple and strong baseline for collaborative filtering. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 1243–1252.

[22] Desnes Nunes, Ricardo Primi, Ramon Pires, Roberto de Alencar Lotufo, and Rodrigo Nogueira. 2023. Evaluating GPT-3.5 and GPT-4 Models on Brazilian University Admission Exams. *ArXiv* abs/2303.17003 (2023).

[23] OpenAI. 2022. ChatGPT: Optimizing language models for dialogue. https://openai.com/blog/âĂŒchatgpt/

[24] Yoon-Joo Park and Alexander Tuzhilin. 2008. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM conference on Recommender systems*. 11–18.

[25] Gustavo Penha and Claudia Hauff. 2020. What does bert know about books, movies and music? probing bert for conversational recommendation. In *Proceedings of the 14th ACM Conference on Recommender Systems*. 388–397.

[26] Yujia Qin, Zihan Cai, Dian Jin, Lan Yan, Shihao Liang, Kunlun Zhu, Yankai Lin, Xu Han, Ning Ding, Huadong Wang, et al. 2023. WebCPM: Interactive Web Search for Chinese Long-form Question Answering. *arXiv preprint arXiv:2305.06849* (2023).

[27] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. [n. d.]. Language Models are Unsupervised Multitask Learners. ([n. d.]).

[28] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446* (2021).

[29] Arya S Rao, John Kim, Meghana Kamineni, Michael Pang, Winston Lie, and Marc Succi. 2023. Evaluating ChatGPT as an adjunct for radiologic decision-making. *medRxiv* (2023), 2023–02.

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

[31] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast context-aware recommendations with factorization machines. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*, Wei-Ying Ma, Jian-Yun Nie, Ricardo Baeza-Yates, Tat-Seng Chua, and W. Bruce Croft (Eds.). ACM, 635–644. https://doi.org/10.1145/2009916.2010002

[32] Paul Resnick and Hal R Varian. 1997. Recommender systems. *Commun. ACM* 40, 3 (1997), 56–58.

[33] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100* (2022).

[34] Damien Sileo, Wout Vossen, and Robbe Raymaekers. 2022. Zero-Shot Recommendation as Language Modeling. In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II*. Springer, 223–230.

[35] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990* (2022).

[36] 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*. 1441–1450.

[37] Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agent. *ArXiv* abs/2304.09542 (2023).

[38] Zhongxiang Sun. 2023. A short survey of viewing large language models in legal aspect. *arXiv preprint arXiv:2303.09136* (2023).

[39] Shuohuan Wang, Yu Sun, Yang Xiang, Zhihua Wu, Siyu Ding, Weibao Gong, Shikun Feng, Junyuan Shang, Yanbin Zhao, Chao Pang, Jiaxiang Liu, Xuyi Chen, Yuxiang Lu, Weixin Liu, Xi Wang, Yangfan Bai, Qiuliang Chen, Li Zhao, Shiyong Li, Peng Sun, Dianhai Yu, Yanjun Ma, Hao Tian, Hua Wu, Tian Wu, Wei Zeng, Ge Li, Wen Gao, and Haifeng Wang. 2021. ERNIE 3.0 Titan: Exploring Larger-scale Knowledge Enhanced Pre-training for Language Understanding and Generation. arXiv:2112.12731 [cs.CL]

[40] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Empowering news recommendation with pre-trained language models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1652–1656.

[41] Chen Xu, Jun Xu, Xu Chen, Zhenghua Dong, and Ji-Rong Wen. 2022. Dually Enhanced Propensity Score Estimation in Sequential Recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2260–2269.

[42] Jun Xu, Xiangnan He, and Hang Li. 2018. Deep learning for matching in search and recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 1365–1368.

[43] Yuhui Zhang, Hao Ding, Zeren Shui, Yifei Ma, James Zou, Anoop Deoras, and Hao Wang. 2021. Language models as recommender systems: Evaluations and limitations. (2021).

[44] 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. arXiv:2303.18223 [cs.CL]