Do LLMs Understand User Preferences? Evaluating LLMs On User Rating Prediction

WANG-CHENG KANG*, Google Research, Brain Team, USA
JIANMO NI*, Google Research, Brain Team, USA
NIKHIL MEHTA, Google Research, Brain Team, USA
MAHESWARAN SATHIAMOORTHY, Google Research, Brain Team, USA
LICHAN HONG, Google Research, Brain Team, USA
ED CHI, Google Research, Brain Team, USA
DEREK ZHIYUAN CHENG, Google Research, Brain Team, USA

Large Language Models (LLMs) have demonstrated exceptional capabilities in generalizing to new tasks in a zero-shot or few-shot manner. However, the extent to which LLMs can comprehend user preferences based on their previous behavior remains an emerging and still unclear research question. Traditionally, Collaborative Filtering (CF) has been the most effective method for these tasks, predominantly relying on the extensive volume of rating data. In contrast, LLMs typically demand considerably less data while maintaining an exhaustive world knowledge about each item, such as movies or products. In this paper, we conduct a thorough examination of both CF and LLMs within the classic task of user rating prediction, which involves predicting a user's rating for a candidate item based on their past ratings. We investigate various LLMs in different sizes, ranging from 250M to 540B parameters and evaluate their performance in zero-shot, few-shot, and fine-tuning scenarios. We conduct comprehensive analysis to compare between LLMs and strong CF methods, and find that zero-shot LLMs lag behind traditional recommender models that have the access to user interaction data, indicating the importance of user interaction data. However, through fine-tuning, LLMs achieve comparable or even better performance with only a small fraction of the training data, demonstrating their potential through data efficiency.

ACM Reference Format:
Wang-Cheng Kang*, Jianmo Ni*, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. 2023. Do LLMs Understand User Preferences? Evaluating LLMs On User Rating Prediction. 1, 1 (May 2023), 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Large language models (LLMs) have shown an uncanny ability to handle a wide variety of tasks such as text generation [1, 2, 6, 26], translation [36, 42], and summarization [19]. The recent fine-tuning of LLMs on conversations and the use of techniques like instruction fine-tuning [4] and reinforcement learning from human feedback (RLHF) [3] led to

*The two authors contributed equally to this work.
Authors' addresses: Wang-Cheng Kang*, wckang@google.com, Google Research, Brain Team, USA; Jianmo Ni*, jianmon@google.com, Google Research, Brain Team, USA; Nikhil Mehta, nikhilmehta@google.com, Google Research, Brain Team, USA; Maheswaran Sathiamoorthy, nlogn@google.com, Google Research, Brain Team, USA; Lichan Hong, lichan@google.com, Google Research, Brain Team, USA; Ed Chi, edchi@google.com, Google Research, Brain Team, USA; Derek Zhiyuan Cheng, zcheng@google.com, Google Research, Brain Team, USA.

Manuscript submitted to ACM

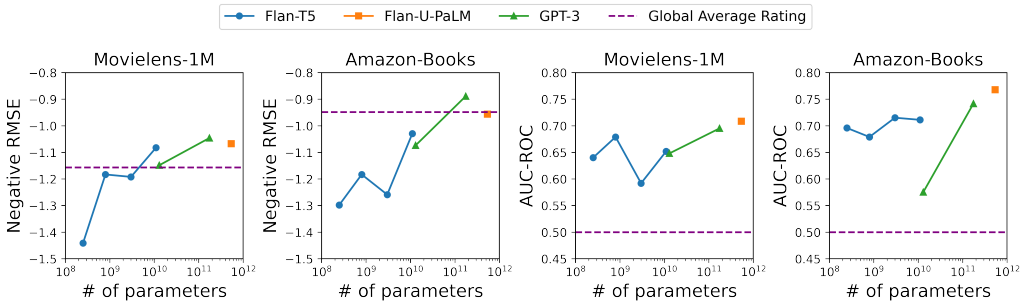


Fig. 1. Zero-shot performance for rating prediction of LLMs with different model sizes, including Flan-T5 (base to XXL), GPT3 (Curie, Davinci) and Flan-U-PaLM. We see a performance gain when increasing the model size. Among which, models greater than 100B (Flan-U-PaLM 540B and text-davinci-003 175B) outperform or on-par with the global average rating baseline on both RMSE and AUC-ROC.

tremendous success to bring highly human-like chatbots (e.g. ChatGPT [23], and Bard [10]) to the average household. There are three key factors that directly contribute to LLMs' versatility and effectiveness:

(1) Knowledge from internet-scale real world information: LLMs are trained on enormous datasets of text, providing access to a wealth of real-world information. This information is converted to knowledge that can be used to answer questions, creatives writing (e.g., poems, and articles), and translate between languages.
(2) Incredible generalization ability through effective few-shot learning: within certain context, LLMs are able to learn new tasks from an extremely small number of examples (a.k.a., few-shot learning). Strong few-shot learning capability gears up LLMs to be highly adaptable to new tasks.
(3) Strong reasoning capability: LLMs are able to reason through a chain-of-thought process [40, 41], significantly improving their performance across many tasks [37].

Recently, there has been some early exploratory work to make use of LLMs for Search [20], Learning to Rank [11, 45], and Recommendation Systems [5, 8, 18]. Specifically for recommendation systems, P5 [8] fine-tunes T5-small (60M) and T5-base(220M) [27], unifying both ranking, retrieval and other tasks like summary explanation into one model. M6-Rec [5] tackles the CTR prediction task by finetuning a LLM called M6 (300M) [17]. Liu et al. [18] looked into whether conversational agents like ChatGPT could serve as an off-the-shelf recommender model with prompts as the interface and reported zero-shot performance on rating prediction against baselines like MF and MLP. However, there is a noticeable absence of a comprehensive study that meticulously evaluates LLMs of varying sizes and contrasts them against carefully optimized, strong baselines.

In this paper, we explore the use of off-the-shelf large language models (LLMs) for recommendation systems. We study a variety of LLMs of various sizes ranging from 250M to 540B parameters. We focus on the specific task of user rating prediction, and evaluate the performance of these LLMs under three different regimes: 1. zero-shot 2. few-shot, and 3. fine-tuning. We then carefully compare them with the state-of-the-art recommendation models on two widely adopted recommendation benchmark datasets. Our contributions are three-fold:

- We empirically study the zero-shot and few-shot performance of off-the-shelf LLMs with a wide spectrum of model sizes. We found that larger models (over 100B parameters) can provide reasonable recommendations under the cold-start scenario, achieving comparable performance to decent heuristic-based baselines.

- We show that zero-shot LLMs still fall behind traditional recommender models that utilize human interaction data. Zero-shot LLMs only achieve comparable performance than two surprisingly trivial baselines that always predicts the average item or user rating. Furthermore, they significantly underperform traditional supervised recommendation models, indicating the importance of user interaction data.
- Through numerous experiments that fine-tune LLMs on human interaction data, we demonstrate that fine-tuned LLMs can achieve comparable or even better performance than traditional models with only a small fraction of the training data, showing its promise in data efficiency.

2 RELATED WORK

2.1 Use of Natural Language in Recommender System

One of the earliest works that explored formulating the recommendation problem as a natural language task is [44]. They used BERT [6] and GPT-2 [25] on the Movielens dataset [12] to show that such language models perform surprisingly well, though not as good as well tuned baselines like GRU4Rec [15].

P5 [8] fine-tunes a popular open-sourced T5 [27] model, unifying both ranking, retrieval and other tasks like summary explanation into one model. M6-Rec [5] is another related work, but they tackle the CTR prediction task by finetuning a LLM called M6 [17].

Two recent works explore the use of LLMs for zero-shot prediction. ChatRec [7] handles zero-shot prediction as well as being interactive and providing explanations. [35] takes a three-stage prompting approach to generate next item recommendation in the Movielens dataset and achieves competitive metrics, although not being able to beat strong sequential recommender baselines such as SASRec [16].

2.2 Large Language Models

Once people realized that scaling up sizes of data and model helps language models, there has been a series of large language models proposed and built: e.g. PaLM [2], GPT-3 [1] and recent ones such as OPT [43] and LLaMA [33]. One of the unique abilities of LLMs has been in their ability to reason about things, which is further improved by techniques such as chain-of-thought prompting [41], self-consistency [38] and self-reflection [30].

Another major strong capability of LLMs is instruction following that models can generalize to unseen tasks by following the given natural language instructions. Researchers have found that techniques like instruction fine-tuning [4] and RLHF [3] can significantly improve LLMs' capability to perform tasks given natural language descriptions that align with human's preferences. As one of the tasks that can be described in natural language, 'recommendation' has become a promising new capability for LLMs. In this work, we focus on the models that have been fine-tuned to improve their instruction following capability such as ChatGPT [23], GPT-3 (text-davinci-003 [22]), Flan-U-PaLM and Flan-T5 [4].

3 METHOD

3.1 Problem formulation

We study the task of user rating prediction, formulated as: Given a user $u \in \mathcal{U}$, a sequence of user u's historical interactions $E^u = \{e_1^u, e_2^u, ..., e_n^u\}$ and an item $i \in \mathcal{I}$, predict the rating that the user u will give to the item i, where the user historical interaction sequence E^u is ordered by time (e_n^u is the most recent item that the user consumed), and each

interaction e_k^u is represented by information about the item (e.g., ID, title, metadata, etc.) that the user has consumed as well as the rating the user gave to the item.

3.2 Zero-shot and Few-shot LLMs for Rating Prediction

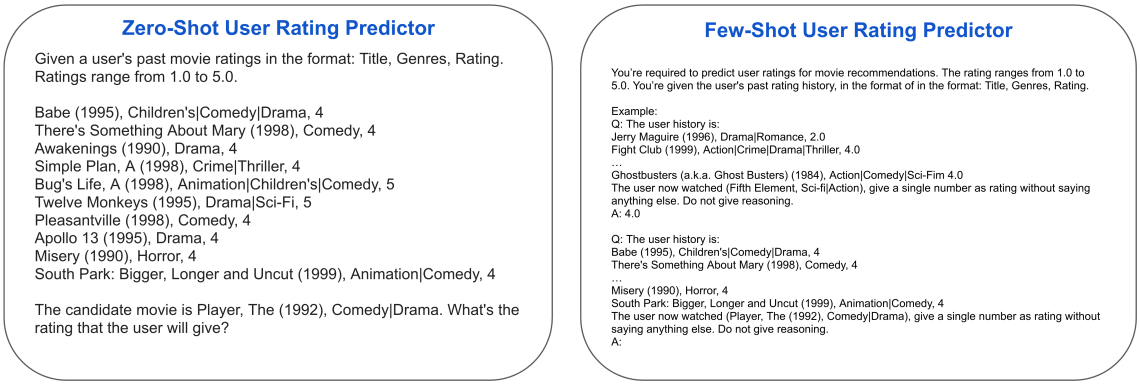


Fig. 2. Zero and few shot LLM prompt for rating prediction.

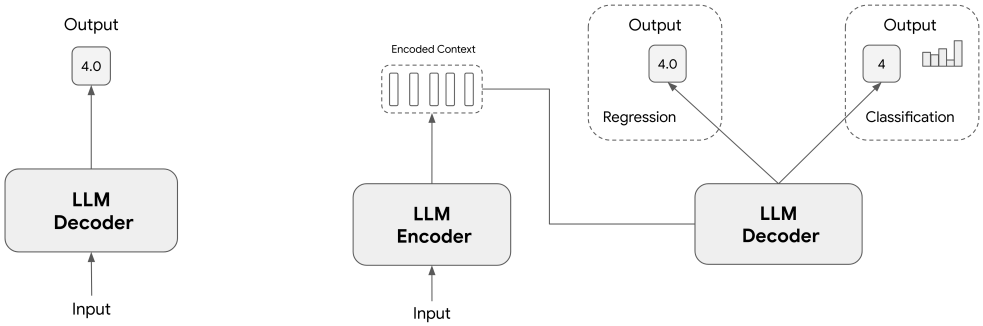
We demonstrate the zero-shot and few-shot prompts used for the rating prediction task on the MovieLens dataset in figure 2. As shown in the figure, the input prompts depict several important features represented as text, including user's past rating history and candidate item features (title and genre). Finally, to elicit a numeric rating from the model with the rating scale, the input prompt specifies a numerical rating scale. The model response is parsed to extract the rating output from the model. However, we discovered that LLMs can be highly sensitive to the input prompts and do not always follow the provided instruction. For instance, we found that certain LLMs may offer additional reasoning or not provide a numerical rating at all. To resolve this, we performed additional prompt engineering by adding additional instructions such as "Give a single number as rating without explanation" and "Do not give reasoning" to the input prompt.

3.3 Fine-tuning LLMs for Rating Prediction

In traditional recommender system research, it has been widely shown that training models with human interaction data is effective and critical to improve recommender's capability of understanding user preference.

Here, we explore training the LLMs with human interaction and study how it could improve the model performance. We focus on fine-tuning a family of LLMs, namely Flan-T5, since they are publicly available and have competitive performance on a wide range of benchmarks. The rating prediction task could be formulated into one of two tasks: (1) multi-class classification; or (2) regression, as shown in Figure 3b.

Multi-class Classification. LLMs (either Decoder-only or Encoder-Decoder architecture) are essentially pre-trained with a K-way classification task that predicts the token from a fixed vocabulary with size K. As shown in Figure 3b, there is a projection layer that projects the outputs from the last layer to the vocabulary size then the pre-training optimizes the cross-entropy loss for the token classification. The output logits is computed as $\text{logits}_{\text{dec}} = W_{\text{proj}} h_{\text{dec}}$, where W_{proj} is the projection matrix of size $(d, |V|)$, h_{dec} is the output from the decoder's last transformer layer, d is the hidden dimension size of the decoder and $|V|$ is the vocabulary.



(a) Decoder-Only model.

(b) Encoder-Decoder model.

Fig. 3. Two types of LLMs for the rating prediction task.

Following [4, 26], we formulate the rating regression task as a 5-way classification task, where we take the rating 1 to 5 as 5 classes. During training, we use the cross-entropy loss as other classification tasks, as shown below:

$$L_{\text{cross\_entropy}} = -\sum_{i=1}^{N} r^i \log(\text{logits}^i_{\text{dec}}), \tag{1}$$

where r^i is the ground-truth rating for the i-th item and N is the number of total training examples.

During inference, we compute the log-likelihood for the model output each class and choose the class with the largest probability as the final prediction.

Regression. . To enable LLMs for regression tasks, we set the shape of the projection matrix W_{proj} to be $(d, 1)$, so that it will only output a 1-digit logits. As shown in Equation 2, during training we apply a mean-squared-error (MSE) loss based on the output logits and the ground-truth rating.

$$L_{\text{regression}} = \frac{1}{|N|} \sum_{i=1}^{N} (\text{logits}^i_{\text{dec}} - r^i)^2. \tag{2}$$

4 EXPERIMENTS

We conduct extensive experiments to answer the following research questions:
RQ1: Do off-the-shelf LLMs perform well for zero-shot and few-shot recommendations?
RQ2: How do LLMs compare with traditional recommenders in a fair setting
RQ3: How much does model size matter for LLMs when used for recommenders?
RQ4: Do LLMs converge faster than traditional recommender models?

4.1 Datasets and Evaluation Setup

4.1.1 Datasets. To evaluate the user rating prediction task, we use two widely adopted benchmark datasets for evaluating model performance on recommendations. Both datasets consist of user review ratings that range from 1 to 5.

- MovieLens [13]: We use the version MovieLens-1M that includes 1 million user ratings for movies.
- Amazon-Books [21]: We use the "Books" category of the Amazon Review Dataset with users' ratings on items. We use the 5-core version that filters out users and items with less than 5 interactions.

Table 1. Statistics of the dataset

Datasets	#Users	#Items	# of Training Examples	# of Test Examples	Features
Movielens-1M	6,040	3,689	882,002	2,000 (75,880)	Title, Genre
Amazon-Books	1,850,187	483,579	17,566,711	2,000 (2,324,503)	Title, Brand

4.1.2 Training / Test Split. To create the training and test sets, we follow the single-time-point split [32]. We first filter out the ratings associated with items that don't have metadata, then sort all user ratings in chronological order. Finally, we take the first 90% ratings as the training set and the remaining as the test set. Each training example is a tuple of $< user\_id, item\_id, item\_metadata, rating >$, where the label is a 5 Likert-scale rating. The input features are $user\_id$, $item\_id$, and a list of item_metadata features. The statistics of the datasets are shown in the Table 1. Due to the high computation cost of zero-shot and few-shot experiments based on LLMs, we randomly sample from the test set of each dataset into 2,000 tuples as a smaller test set. For all our experiments, we report results on the sampled test set. And we truncate the user sequence to the most recent 10 interactions during training and evaluation.

4.1.3 Evaluation Metrcis. We use the widely adopted metrics RMSE (Root Mean Squared Error) and MAE (Mean Average Error) to measure model performance on rating prediction. Moreover, we use ROC-AUC to evaluate the model's performance on ranking, where ratings greater than or equal to 4 are considered as positives and the rest as negatives. In this case, AUC measures whether the model ranks the positives higher than negatives.

4.2 Baselines and LLMs

4.2.1 Baselines.

- **Traditional Recommeder**: We consider several traditional recommendation models as strong baselines, including 1. Matrix Factorization (**MF**) [29], and 2. Multi-layer Perceptrons (**MLP**) [14]. For MF and MLP, only user ID and item ID are used as input features.
- **Attribute and Rating-aware Sequential Rating Predictor**: In our experiments, we supply the LLM with historical item metadata, such as titles and categories, along with historical ratings. However, to the best of our knowledge, there is no existing method designed for this setting[1]. To ensure a fair comparison, we construct a **Transformer-MLP** model to efficiently process the same input information provided to the LLM.

  There are three key design choices: (i) feature processing: We treat all features as sparse features, and learn their embeddings end-to-end. For example, we use one-hot encoding for genres, and create an embedding table, where the i-th row is genre i's embedding. Similarly, we obtain bag-of-words encodings via applying a tokenizer[2] on titles, and then look up the corresponding embedding. (ii) user modeling: for each user behavior, we use `Add` or `Concat` to aggregate all embeddings (e.g. item ID, title, genres/category, rating) into one, and then adopt bi-directional self-attention [34] layers with learned position embeddings to model users' past behaviors. Similar to SASRec, we use the most recent behavior's output embedding as the user summary; (iii) Fuse user and candidate for rating prediction: we apply a MLP on top of the user embedding along with other candidate item features to generate the final rating prediction, and optimize for minimizing MSE.

[1]The most related works are SASRec [16] and CARCA [28], however they are designed for next item prediction instead of rating prediction, and thus not directly applicable to our case.
[2]https://www.tensorflow.org/text/api_docs/python/text/WhitespaceTokenizer

Table 2. User rating prediction results. The best performing method is boldfaced in each column, and underlined in each group.

Model	MovieLens			Amazon-Books		
	RMSE↓	MAE↓	AUC↑	RMSE↓	MAE↓	AUC↑
Zero-Shot LLMs						
Flan-U-PALM	1.0677	0.7740	0.7084	0.9565	0.5569	0.7676
ChatGPT	1.0081	0.8193	0.6794	1.0081	0.8093	0.6778
text-davinci-003	1.0460	0.7850	0.6951	0.8890	0.5442	0.7416
Few-Shot LLMs						
Flan-U-PALM	1.0721	0.7605	0.7094	1.0712	0.5855	0.7439
ChatGPT	1.0862	0.8203	0.6930	1.0618	0.7760	0.7470
text-davinci-003	1.0867	0.8119	0.6963	1.0716	0.7753	0.7739
Simple Dataset Statistics						
Global Avg. Rating	1.1564	0.9758	0.5	0.9482	0.7609	0.5
Candidate Item Avg. Ratings	0.9749	0.7778	0.7395	0.9342	0.7078	0.6041
User Past Avg. Ratings	1.0196	0.7959	0.7266	0.8527	0.5502	0.8047
Supervised Recommendation Methods						
MF	0.9552	0.7436	0.7734	1.7960	1.1070	0.7638
MLP	0.9689	0.7452	0.7393	0.8607	0.6384	0.6932
Transformer+MLP	**0.8848**	0.7036	0.7979	**0.8143**	0.5541	0.8042
Fine-tuned LLMs						
Flan-T5-Base (classification)	1.0110	0.6805	0.7590	0.9856	0.4685	0.6292
Flan-T5-Base (regression)	0.9187	0.7092	0.7949	0.8413	0.5317	0.8182
Flan-T5-XXL (regression)	0.8979	**0.6986**	**0.8042**	0.8301	**0.5122**	**0.8312**

To properly tune the baseline models, we defined a hyper-parameter search space (e.g. for embedding dimension, learning rate, network size, `Add` or `Concat` aggregation, etc.), and perform more than 100 search trials using Vizier [9], a black-box hyper-parameter optimization tool.

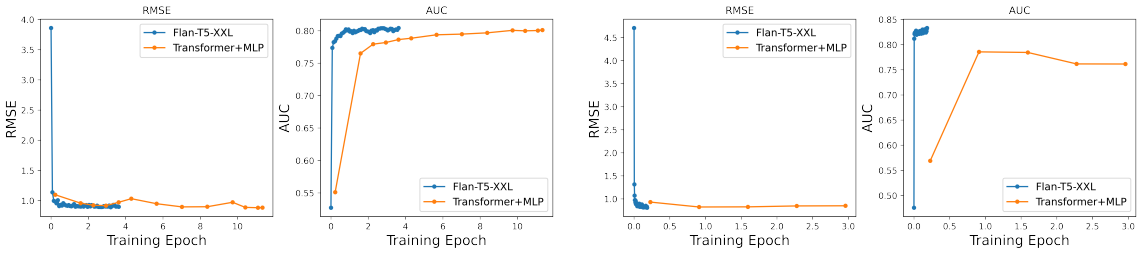
- **Heuristics**: We also include three heuristic-based baselines: (1) global average rating: (2) candidate item average rating, and (3) user past average rating, meaning the model's prediction is depending on (1) the average rating among all user-item ratings, (2) the average rating from the candidate item or (3) the user's average rating in the past.

4.2.2 LLMs for Zero-shot and Few-shot Learning. : We used the LLMs listed below for zero-shot and few-shot learning. We use a temperature of 0.1 for all LLMs, as the LLM's output in our case is simply a rating prediction. We use GPT-3 models from OpenAI [24]: (i) **text-davinci-003 (175B)**: The most capable GPT-3 model with Reinforcement Learning from Human Feedback (RLHF) [31]; (ii) **ChatGPT**: the default model is gpt-3.5-turbo, fine-tuned on both human-written demonstrations and RLHF, and further optimized for conversation. **Flan-U-PaLM (540B)** is the largest and strongest model in [4], it applies both FLAN instruction tuning [39] and UL2 training objective [4] on PaLM [2].

4.2.3 LLMs for Fine-tuning. For fine-tuning methods, we use **Flan-T5-Base** (250M) and **Flan-T5-XXL** (11B) models in the experiments. We set the learning rate to 5e-5, batch size to 64, drop out rate to 0.1 and train 50k steps on all datasets.

4.3 Zero-Shot and Few-shot LLMs (RQ1)

As shown in Table 2, we conduct experiments on several off-the-shelf LLMs in the zero-shot setting. We observed that LLMs seem to understand the task from the prompt description, and predict reasonable ratings. LLMs outperform global average rating in most cases, and perform comparable with item or user average ratings. For example, text-davinci-003 performs slightly worse than candidate item average ratings on Movielens but outperforms on Amazon-Books. For



(a) MovieLens (b) Amazon Books

Fig. 4. Data efficiency: convergence curve.

few-shot experiments, we provide 3 examples in the prompt (3-shot). Compared against zero-shot, we found that the AUC for few-shot LLMs are improved, while there is no clear pattern in RMSE and MAE.

Furthermore, we found that both zero-shot and few-shot LLMs under-perform traditional recommendation models trained with interaction data. As shown in Table 2, GPT-3 and Flan-U-PaLM models achieve significantly lower performance compared to supervised models. The inferior performance could be due to the lack of user-item interaction data in LLMs' pre-training, and hence they don't have knowledge about human preference for different recommendation tasks. Moreover, recommendation tasks are highly dataset-dependent: (e.g.) the same movie can have different average ratings on different platform. Hence, without knowing the dataset-specific statistics, it's impossible for LLM to provide a universal prediction that is suitable for every datasets.

4.4 LLMs vs. Traditional Recommender Models (RQ2)

Fine-tuning LLMs is an effective way to feed dataset statistics into LLMs, and we found the performance of fine-tune LLMs are much better than zero/few-shot LLMs. Also, when fine-tuning the Flan-T5-base model with the classification loss, the performance is much worse than fine-tuning with the regression loss on all three metrics. This indicates the importance of choosing the right optimizing objective for fine-tuning LLMs.

Comparing against the strongest baseline Transformer-MLP, we found fine-tuned Flan-T5-XXL has better MAE and AUC, implying fine-tune LLMs may be more suitable for ranking tasks.

4.5 Effect of Model Size (RQ3)

For all the LLMs we studied of different model sizes vary from 250M and 500B parameters, we were able to use zero-shot or few-shot prompts to let them output a rating prediction between 1 to 5. This shows the effectiveness of instruction tuning that enables these LLMs (Flan-T5, Flan-U-PaLM, GPT-3) to follow the prompt. We further found that only LLMs with size greater than 100B perform reasonably well on rating prediction in the zero-shot setting, as shown in Figure 1. For fine-tuning experiments, we also found that Flan-T5-XXL outperforms Flan-T5-Base on both datasets, as shown in the last two rows of Table 2.

4.6 Data Efficiency of LLMs (RQ4)

As LLMs have learned vast amounts of world knowledge during pre-training, while traditional recommender models are trained from scratch, we compare their convergence curves in Figure 4 to examine whether LLMs have better data efficiency. We can see that for RMSE, both methods could converge to reasonable performance with a small fraction of

data. This is probably because that even average rating of all items has a relatively low RMSE, and thus as long as a model learns to predict a rating near the average rating, it could achieve reasonable performance. For AUC the trend is more clear, as simply predicting average rating results in an AUC of 0.5. We found that a small fraction of data is required for LLM to achieve good performance, while Transformer+MLP needs much more training data (at least 1 epoch) for convergence.

5 CONCLUSION

In this paper, we evaluate the effectiveness of large language models as a recommendation system for user rating prediction in three settings: 1. zero-shot; 2. few-shot; and 3. fine-tuning. Compared to traditional recommender methods, our results revealed that LLMs in zero-shot and few-shot LLMs fall behind fully supervised methods, implying the importance of incorporating the target dataset distribution into LLMs. On the other hand, fine-tuned LLMs can largely close the gap with carefully designed baselines in key metrics. LLM-based recommenders have several benefits: (i) better data efficiency; (ii) simplicity for feature processing and modeling: we only need to convert information into a prompt without manually designing feature processing strategies, embedding methods, and network architectures to handle various kind of information; (iii) potential for unlock conversational recommendation capabilities. Our work sheds light on the current status of LLM-based recommender systems, and in the future we will further look into improving the performance via methods like prompt tuning, and explore novel recommendation applications enabled by LLMs.

REFERENCES

[1] 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.

[2] 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).

[3] Paul Francis Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep Reinforcement Learning from Human Preferences. *ArXiv* abs/1706.03741 (2017).

[4] Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed Huai hsin Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling Instruction-Finetuned Language Models. *ArXiv* abs/2210.11416 (2022).

[5] 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).

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-REC: Towards Interactive and Explainable LLMs-Augmented Recommender System. *arXiv preprint arXiv:2303.14524* (2023).

[8] 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). *arXiv preprint arXiv:2203.13366* (2022).

[9] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D. Sculley. 2017. Google Vizier: A Service for Black-Box Optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. ACM, 1487–1495. https://doi.org/10.1145/3097983.3098043

[10] Google. 2023. Bard: A Large Language Model from Google AI. https://bard.google.com/

[11] Shuguang Han, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2020. Learning-to-Rank with BERT in TF-Ranking. arXiv:2004.08476 [cs.IR]

[12] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[13] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5 (2016), 19:1–19:19.

[14] 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, WWW 2017, Perth, Australia, April 3-7, 2017*, Rick Barrett, Rick Cummings, Eugene Agichtein, and Evgeniy Gabrilovich (Eds.). ACM, 173–182. https://doi.org/10.1145/3038912.3052569

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

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

[17] Junyang Lin, Rui Men, An Yang, Chang Zhou, Ming Ding, Yichang Zhang, Peng Wang, Ang Wang, Le Jiang, Xianyan Jia, et al. 2021. M6: A chinese multimodal pretrainer. *arXiv preprint arXiv:2103.00823* (2021).

[18] Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is ChatGPT a Good Recommender? A Preliminary Study. arXiv:2304.10149 [cs.IR]

[19] Yang Liu and Mirella Lapata. 2019. Text Summarization with Pretrained Encoders. arXiv:1908.08345 [cs.CL]

[20] Microsoft. 2023. Reinventing search with a new AI-powered Bing and Edge, your copilot for the web. https://news.microsoft.com/the-new-Bing/

[21] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Conference on Empirical Methods in Natural Language Processing*.

[22] OpenAI. 2022. Aligning language models to follow instructions. https://openai.com/research/instruction-following.

[23] OpenAI. 2022. Introducing ChatGPT. https://openai.com/blog/chatgpt.

[24] OpenAI. 2023. GPT Models Documentation. https://platform.openai.com/docs/models/overview

[25] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[26] 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. *J. Mach. Learn. Res.* 21, 1, Article 140 (jan 2020), 67 pages.

[27] 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 (2020), 5485–5551.

[28] Ahmed Rashed, Shereen Elsayed, and Lars Schmidt-Thieme. 2022. CARCA: Context and Attribute-Aware Next-Item Recommendation via Cross-Attention. *arXiv preprint arXiv:2204.06519* (2022).

[29] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[30] Noah Shinn, Beck Labash, and Ashwin Gopinath. 2023. Reflexion: an autonomous agent with dynamic memory and self-reflection. *arXiv preprint arXiv:2303.11366* (2023).

[31] Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. Learning to summarize from human feedback. *CoRR* abs/2009.01325 (2020). arXiv:2009.01325 https://arxiv.org/abs/2009.01325

[32] Aixin Sun. 2022. Take a Fresh Look at Recommender Systems from an Evaluation Standpoint.

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

[34] 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 (2017).

[35] Lei Wang and Ee-Peng Lim. 2023. Zero-Shot Next-Item Recommendation using Large Pretrained Language Models. *arXiv preprint arXiv:2304.03153* (2023).

[36] Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. 2019. Learning Deep Transformer Models for Machine Translation. arXiv:1906.01787 [cs.CL]

[37] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. arXiv:2203.11171 [cs.CL]

[38] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171* (2022).

[39] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned Language Models are Zero-Shot Learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net. https://openreview.net/forum?id=gEZrGCozdqR

[40] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent Abilities of Large Language Models. arXiv:2206.07682 [cs.CL]

[41] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903* (2022).

[42] Jiacheng Yang, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Weinan Zhang, Yong Yu, and Lei Li. 2020. Towards making the most of bert in neural machine translation. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 9378–9385.

[43] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068* (2022).

[44] 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).

[45] Lixin Zou, Shengqiang Zhang, Hengyi Cai, Dehong Ma, Suqi Cheng, Shuaiqiang Wang, Daiting Shi, Zhicong Cheng, and Dawei Yin. 2021. Pre-Trained Language Model Based Ranking in Baidu Search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 4014–4022. https://doi.org/10.1145/3447548.3467147