

Do LLMs Memorize Recommendation Datasets?

A Preliminary Study on MovieLens-1M

Dario Di Palma*
d.dipalma2@phd.poliba.it
Politecnico di Bari
Bari, Italy

Felice Antonio Merra*
felice.merra@cognism.com
Cognism
Remote, Italy

Maurizio Sfilio
m.sfilio@studenti.poliba.it
Politecnico di Bari
Bari, Italy

Vito Walter Anelli
vitowalter.aneli@poliba.it
Politecnico di Bari
Bari, Italy

Fedelucio Narducci
fedelucio.narducci@poliba.it
Politecnico di Bari
Bari, Italy

Tommaso Di Noia
tommaso.dinoia@poliba.it
Politecnico di Bari
Bari, Italy

Abstract

Large Language Models (LLMs) have become increasingly central to recommendation scenarios due to their remarkable natural language understanding and generation capabilities. Although significant research has explored the use of LLMs for various recommendation tasks, little effort has been dedicated to verifying whether they have memorized public recommendation dataset as part of their training data. This is undesirable because memorization reduces the generalizability of research findings, as benchmarking on memorized datasets does not guarantee generalization to unseen datasets. Furthermore, memorization can amplify biases, for example, some popular items may be recommended more frequently than others.

In this work, we investigate whether LLMs have memorized public recommendation datasets. Specifically, we examine two model families (GPT and Llama) across multiple sizes, focusing on one of the most widely used dataset in recommender systems: MovieLens-1M. First, we define dataset memorization as the extent to which item attributes, user profiles, and user-item interactions can be retrieved by prompting the LLMs. Second, we analyze the impact of memorization on recommendation performance. Lastly, we examine whether memorization varies across model families and model sizes. Our results reveal that all models exhibit some degree of memorization of MovieLens-1M, and that recommendation performance is related to the extent of memorization. We have made all the code publicly available at: [GitHub](#)

Keywords

Large Language Models (LLMs), Dataset Memorization, Recommender Systems

*Corresponding author.

This is the authors' version of the work. The final, published version will appear in the *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25)*.

This work is licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0).

Please cite the official published version when available.

Conference'17, July 2017, Washington, DC, USA
2025. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
<https://doi.org/10.1145/nnnnnnnn.nnnnnnnn>

1 Introduction

Large Language Models (LLMs) have emerged as versatile tools in Recommender Systems (RSs), leveraging their extensive world-knowledge and reasoning capabilities [15]. Currently, these models are utilized in three primary ways: fine-tuning [3, 28, 33, 46], prompt-based methods [10, 11, 18, 30], and data augmentation [4, 34, 37, 38].

Beyond these methodologies, the successful integration of LLMs into RSs has already led to impactful applications. For example, these models have been used in knowledge augmentation [35, 37, 38], LLM-enhanced recommendation [17, 31, 40], and even as standalone recommenders [12, 21, 44, 46].

Although research efforts have focused primarily on designing solutions to improve recommendation tasks through LLM, limited attention has been paid to understanding the underlying reasons for their effectiveness. Among the various factors that may contribute to these advantages, a key aspect is addressing the question: *To what extent have these models memorized the dataset during training?*

In related fields, researchers have begun to address this question by defining and quantifying LLM memorization. For example, Carlini et al. [6] found that the GPT-J-6B model memorized at least 1% of the Pile dataset [14], while Al-Kaswan et al. [2] were able to extract 56% of the coding samples used to train GPT-Neo.

In this work, we aim to answer the question: *Do LLMs Memorize Recommendation Datasets?* We believe that addressing this question is fundamental, as the memorization of recommendation datasets can lead to several issues specific to the recommendation community. Potential issues include: (i) **Non-generalizable test results**, i.e., metrics computed on test datasets that have been memorized, lead to unreliable performance. (ii) **Amplification of biases**, i.e., if a memorized dataset exhibits popularity bias, the LLM may over-recommend popular items at the expense of less popular ones. (iii) **Unfair comparison with standard recommender systems**, i.e., non-LLM-based recommenders do not inherently possess cross-domain knowledge.

In this work, we provide a preliminary answer to the previous question by investigating the memorization of MovieLens-1M, one of the most popular dataset used to evaluate recommender systems [32]. Our main contributions can be summarized as follows: (i) Define memorization in the context of recommendation datasets. (ii) Develop a prompt-based method to probe LLMs for extracting memorized information. (iii) Assess to what extent the Llama and GPT

model families have memorized MovieLens-1M. (iv) Investigate the impact of MovieLens-1M memorization on recommendation tasks. (v) Investigate whether LLMs memorize the characteristics of the dataset, including biases, and whether larger models exhibit higher memorization.

2 Methodology

2.1 Definitions of Memorization

Let a dataset \mathcal{D} be defined as $\mathcal{D} = \{\mathcal{I}, \mathcal{U}, \mathcal{U-I}\}$ where:

$\mathcal{I} = \{(id_i, \text{attributes}_i)\}_{i=1}^m$, is the set of item metadata, with id_i the unique identifier for item i , and attributes_i the metadata associated with item i .

$\mathcal{U} = \{(id_u, \text{attributes}_u)\}_{u=1}^n$, is the set of user metadata, where id_u the unique identifier for user u , and attributes_u the metadata associated with user u .

$\mathcal{U-I} = \{(id_u, id_i, r_{ui})\}_{i=1}^p$, representing the user-item interaction history, with id_u a fixed user ID, id_i the item ID, and r_{ui} the interaction data (e.g., rating, click, etc.).

DEFINITION 1 (ITEM MEMORIZATION). Given an item ID $id_i \in \mathcal{I}$, an LLM is said to memorize the item if it can produce the associated attributes_i .

$$LLM(P_I(id_i)) = \text{attributes}_i$$

where P_I is a prompt designed to query the LLM about item id_i .

DEFINITION 2 (USER MEMORIZATION). A user $id_u \in \mathcal{U}$ is said to be memorized by an LLM if the model can produce the corresponding attributes_u when prompted.

$$LLM(P_U(id_u)) = \text{attributes}_u$$

where P_U is a prompt designed to query the LLM about user id_u .

DEFINITION 3 ((USER-ITEM) INTERACTION MEMORIZATION). Given a subset of user-item interactions $\mathcal{U-I}_{id_u} \subset \mathcal{U-I}$ associated with a fixed user id_u , an LLM is said to memorize an interaction if, given the previous k interactions and the fixed id_u , it correctly predicts (id'_i, r'_{ui}) in the subsequent interaction. Formally, this is expressed as:

$$LLM(P_{\mathcal{U-I}}(\mathcal{U-I}_{id_u})) = (id_u, id'_i, r'_{ui}),$$

where $P_{\mathcal{U-I}}$ is a prompt designed to query the LLM about interaction, id_u is the fixed user identifier, id'_i is a new item identifier, and r'_{ui} represents the predicted interaction.

2.2 Evaluation Protocol

The Dataset. To select the dataset for analysis, we examined the papers accepted at ACM RecSys 2024 [22] and ACM SIGIR 2024 [39] and found that MovieLens-1M¹ was the most frequently used, appearing in 17.2% and 22% of them. This finding aligns with previous research by Vente et al. [32], which reported similar trends in ACM RecSys 2023 [43].

The dataset is composed by three raw files:

- (i) Movies.dat, which stores item data in the format `MovieID::Title::Genres`; (ii) Users.dat, which returns user profiles formatted as `UserID::Gender::Age::Occupation::Zip-code`; and (iii) Ratings.dat,

¹<https://grouplens.org/datasets/movielens/1m/>

System: "You are the MovieLens1M dataset. When given a lookup key (e.g., a MovieID), you will respond with the exact corresponding value from the dataset. Only respond with the value itself. If the key is unknown, respond with 'Unknown'. Below are examples of queries and their correct responses. Follow this pattern strictly. Let's think step by step."
User: "Input: example['ID']::"
Assistant: "example['ID']::example['RealValues']"
User: "Input: example['ID']::"
Assistant: "example['ID']::example['RealValues']"
User: "Input: 'ID'::"

Figure 1: Few-Shot Prompting for Item/User Data Extraction

System: "You are a Recommender Systems. Continue user-item interactions list providing the next interaction based on the MovieLens1M dataset. When given 'UserID::CurrentInteraction', respond with 'UserID::NextInteraction'. Below are examples of queries and their correct responses. Follow this pattern strictly. Let's think step by step."
User: "example['UserID']::"
Assistant: "example['UserID']::example['MovieID']"
User: "example['UserID']::"
Assistant: "example['UserID']::example['MovieID']"
User: "'UserID'::"

Figure 2: Few-Shot Prompting for Interactions Data Extraction

which contains the recorded interactions structured as `UserID::MovieID::Rating::Timestamp`.

Extraction Technique. Data extraction from LLMs is a research area in which emerging techniques rely on prompting. Carlini et al. [7] employed zero-shot prompting to conduct the first adversarial attacks assessing data memorization in GPT-2. Similarly, Yu et al. [41] used a prefix-based prompt to evaluate how much information GPT-Neo memorized.

Inspired by these works, we investigated hand-engineered zero-shot [24], few-shot [5], and Chain-of-Thought [36] prompting techniques. We identified few-shot prompting as the optimal approach for extracting the MovieLens-1M dataset from LLMs. Although finding the best prompt through techniques such as automatic prompt engineering [45] could improve performance, we leave this exploration to future work. The prompts used in this work are shown in Figures 1 and 2.

Metrics. To quantify the memorization of a recommendation dataset, we define three coverage-based metrics.

DEFINITION 4 (MEMORIZATION COVERAGE). Given the set of items \mathcal{I} , the Items' Memorization Coverage (Cov_I) is defined as:

$$Cov(\mathcal{I}, P_I) = \frac{|M(\mathcal{I}, P_I)|}{|\mathcal{I}|} \quad (1)$$

where $M(\mathcal{I}, P_I) = \{id_i \in \mathcal{I} \mid LLM(P_I(id_i)) = \text{attributes}_i\}$ is the subset of items memorized by the LLM using the prompt P_I .

Table 1: Coverage of movies.dat, users.dat, and ratings.dat. Models are grouped by version and ordered by size.

Model Name	Item Coverage (3883 items)	User Coverage (6040 users)	Interaction Coverage (1M interactions)
GPT-4o	80.76%	16.52%	9.37%
GPT-4o mini	8.47%	13.34%	7.17%
GPT-3.5 turbo	60.47%	17.38%	8.92%
Llama-3.3 70B	7.65%	5.84%	2.08%
Llama-3.2 3B	2.68%	13.26%	6.22%
Llama-3.2 1B	1.93%	10.98%	6.49%
Llama-3.1 405B	15.09%	15.30%	8.32%
Llama-3.1 70B	8.01%	15.81%	6.83%
Llama-3.1 8B	3.71%	13.59%	3.82%

Similarly, we define the Users’ Memorization Coverage as $Cov(\mathcal{U}, P_{\mathcal{U}}) = \frac{|M(\mathcal{U}, P_{\mathcal{U}})|}{|\mathcal{U}|}$ and the Interaction Memorization Coverage as $Cov(\mathcal{R}, P_{\mathcal{U}-I}) = \frac{|M(\mathcal{R}, P_{\mathcal{U}-I})|}{|\mathcal{R}|}$ where \mathcal{R} is the set of user-item interactions rows stored in the dataset.

Analyzed LLMs. We conducted our experiments on two families of LLMs widely used in the recommendation community (i.e., Llama [13] and GPT [23]). For the Llama family we experimented with Llama-3.3 70B, Llama-3.2 3B, Llama-3.2 1B, Llama-3.1 405B, Llama-3.1 70B, and Llama-3.1 8B, while, for the GPT family we experimented with GPT-4o, GPT-4o mini, and GPT-3.5 turbo. We study multiple models with different numbers of parameters to investigate whether model size affects the memorization metrics.

To ensure consistent and deterministic behavior across all models, we configured the temperature to 0, prioritizing the most likely token at each step. We set top_p to 1 to include all possible tokens and disabled both frequency_penalty and presence_penalty by setting them to 0. Additionally, we fixed the random seed to 42 to ensure that all stochastic processes behaved consistently across runs.

3 Results and Discussion

3.1 Analysis of Memorization

Items’ Memorization. To examine the extent of LLMs’ memorization of MovieLens-1M items, we used the prompt in Figure 1 and queried LLMs for exact MovieID::Title matches. Table 1 summarizes the item coverage results.

Among the models, GPT-4o achieved the highest coverage, recovering 80.76% of the items, followed by GPT-3.5 turbo with 60.47% coverage. Among open-source models, only Llama-3.1 405B achieved moderate coverage, retrieving 15.09% of the items. Other models, including GPT-4o mini and the smaller Llama variants, retrieved significantly fewer items, with Llama-3.2 1B achieving the lowest coverage at just 1.93%.

Users’ Memorization. To evaluate the extent to which LLMs memorize user attributes, we used the prompt shown in Figure 1 and queried models for exact matches of UserID::Gender::Age::Occupation::Zip-code. Results are shown in Table 1.

GPT-3.5 turbo demonstrated the highest coverage, correctly recovering attributes for 17.38% of users, followed closely by GPT-4o with 16.52% coverage. Among open-source models, Llama-3.1 70B

System: "You are a movie recommendation system for the MovieLens-1M dataset. Based on the user’s past interactions, generate a ranked list of exactly 50 new movie recommendations. Your output must contain only the list in the following format: one line per recommendation in the exact format 'Rank. Title' (e.g., '1. Harry Potter'). Do not include any additional text, commentary, or explanation."

User: "User {user_id} has interacted with the following movies: {training_history_str}. Based solely on these interactions, please generate a ranked list of exactly 50 movie recommendations. Output only the list with no additional commentary or explanation. Each recommendation must be on a new line in the exact format: 'Rank. Title' (for example: '1. Harry Potter')."

Figure 3: Zero-Shot Prompting for Recommendation Task

performed best, achieving 15.81% coverage, while Llama-3.3 70B lagged behind, recovering attributes for only 5.84% of users.

Interaction Memorization. To assess the models’ memorization of user-item interactions, we used the prompt shown in Figure 2 and queried models for exact matches of UserID::MovieID. Table 1 summarizes the results.

GPT-4o achieved the highest coverage, recovering 9.37%, followed by GPT-3.5 turbo, which recovered 8.02% of interactions. Within the Llama family, Llama-3.1 405B achieved 8.32% coverage, followed by Llama-3.1 70B with 6.83%, while Llama-3.3 8B trailed with only 3.82%.

Observation 1. Our findings demonstrate that LLMs possess extensive knowledge of the MovieLens-1M dataset, covering items, user attributes, and interaction histories. Notably, a simple prompt enables GPT-4o to recover nearly 80% of MovieID::Title records. None of the examined models are free of this knowledge, suggesting that MovieLens-1M data is likely included in their training sets. We observed similar trends in retrieving user attributes and interaction histories.

3.2 Impact on Recommendation

To investigate the impact of memorization on recommendation tasks, we evaluate LLMs when prompted to act as a recommender system. The prompt is shown in Figure 3. We ground our experimental results on well-known RS and report the performance of UserKNN [26], ItemKNN [19], BPRMF [25], EASE^R [29], LightGCN [16], MostPop, and Random. The results in Table 2 are computed on MovieLens-1M without any filtering, splitting the dataset into 80% for training and 20% for testing, using the leave-n-out paradigm, following [8, 9, 42].

Overall, LLMs demonstrate strong capabilities in recommendation tasks. Notably, smaller models achieve performance comparable to the strongest baseline. For instance, Llama 3B attains an HR@1 of 0.0421, outperforming its 1B variant (0.0222 HR@1). Meanwhile, larger models such as GPT-4o and Llama 405B surpass both smaller models and baselines (BPRMF, HR@1 = 0.0406). Specifically, GPT-4o, with an HR@1 of 0.2796, outperforms GPT-4o mini (0.0316 HR@1), while Llama 405B achieves an HR@1 of 0.1975, compared to 0.0687 HR@1 for Llama 8B. These trends generalize across all cutoff values.

Table 2: Recommendation accuracy performance on standards recommended and LLM-based recommendation. LLMs are grouped by model family and sorted by size. Best performance in each family is shown in **bold**.

Model Name	HR@1	nDCG@1	HR@5	nDCG@5	HR@10	nDCG@10
Random	0.0093	0.0093	0.0442	0.0092	0.0851	0.0094
MostPop	0.0212	0.0212	0.0775	0.0222	0.1520	0.0251
UserKNN	0.0306	0.0306	0.1209	0.0306	0.2250	0.0347
ItemKNN	0.0394	0.0394	0.1217	0.0353	0.1828	0.0337
BPRMF	0.0406	0.0406	0.1278	0.0350	0.2149	0.0356
EASE ^R	0.0295	0.0295	0.1124	0.0278	0.1975	0.0299
LightGCN	0.0358	0.0358	0.1136	0.0306	0.1882	0.0311
GPT-4o	0.2796	0.2796	0.5889	0.2276	0.6897	0.1948
GPT-4o mini	0.0316	0.0316	0.2132	0.0451	0.3091	0.0413
GPT-3.5 turbo	0.2298	0.2298	0.4217	0.1281	0.5902	0.1229
Llama-3.3 70B	0.2293	0.2293	0.4985	0.1693	0.5922	0.1359
Llama-3.2 3B	0.0421	0.0421	0.1886	0.0443	0.2982	0.0432
Llama-3.2 1B	0.0222	0.0222	0.1018	0.0234	0.1419	0.0207
Llama-3.1 405B	0.1975	0.1975	0.4165	0.1294	0.5119	0.1039
Llama-3.1 70B	0.1302	0.1302	0.3828	0.1095	0.5148	0.0969
Llama-3.1 8B	0.0687	0.0697	0.2281	0.0609	0.3500	0.0571

Observation 2. Although the recommendation performance appears outstanding, comparing Table 2 with Table 1 reveals an interesting pattern. Within each group, the model with higher memorization also demonstrates superior performance in the recommendation task. For example, GPT-4o outperforms GPT-4o mini, and Llama-3.1 405B surpasses Llama-3.1 70B and 8B. These results highlight that evaluating LLMs on datasets leaked in their training data may lead to overoptimistic performance, driven by memorization rather than generalization.

3.3 Impact of Model Scale

Previous analysis highlights that GPT and Llama models are affected by the memorization of the MovieLens-1M dataset and demonstrates that models with higher memorization tend to achieve stronger performance in recommendation tasks. Additionally, a comparison of open-source models with declared sizes shows that larger models not only perform better but also memorize more of the dataset.

For example, Llama-3.1 405B exhibits a mean memorization rate of 12.9%, compared to Llama-3.1 8B with 5.82%, representing a ↓54.88% reduction in memorization. This reduction results in an average decrease of ↓54.23% in nDCG and ↓47.36% in HR (from 405B to 8B).

Observation 3. These findings suggest that increasing the model scale leads to greater memorization of the dataset, resulting in improved performance. Consequently, while larger models exhibit better recommendation performance, they also pose risks related to potential leakage of training data.

3.4 Popularity Memorization

Additionally, we investigated whether these models have also memorized biases inherent in the MovieLens-1M dataset. Specifically, we focused on *popularity bias* [1], examining whether LLMs are more likely to memorize popular items compared to less popular ones. We created three subsets by selecting (i) the top 20% most popular items, (ii) the bottom 20% least popular items with few interactions, and

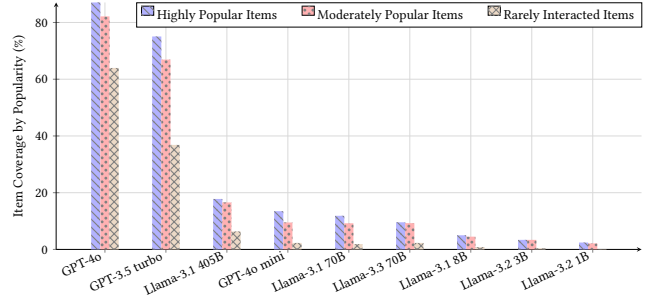


Figure 4: Comparison of item coverage across models by popularity tier. The figure shows the percentage of items covered in three categories: Highly Popular (Top 20%), Moderately Popular (Middle 20%), and Rarely Interacted (Bottom 20%).

(iii) a sample of moderately popular items from the middle of the distribution. We analyzed the coverage of the retrieved items across these subsets and present the results in Figure 4. Overall, larger LLMs exhibit superior performance in retrieving popular items compared to smaller models. For instance, GPT-4o retrieves 89.06% of highly popular items, 86.68% of moderately popular items, and 63.97% of rarely interacted items, whereas GPT-4o mini achieves only 13.48%, 9.49%, and 2.29%. A similar trend is observed in the Llama family, where Llama-3.1 405B attains 19.79%, 18.00%, and 6.34%, while Llama-3.1 8B reaches 4.99%, 4.50%, and 3.86%.

Observation 4. Our findings reveal a pronounced popularity bias in LLMs, with the top 20% of popular items being significantly easier to retrieve than the bottom 20%.

This trend highlights the influence of the training data distribution, where popular movies are overrepresented, leading to their disproportionate memorization by the models.

4 Conclusion and open directions

In this work, we systematically investigate GPT and Llama models to assess whether the MovieLens-1M dataset has been memorized. Our findings reveal that a substantial portion of the item catalog, user metadata, and user interactions can be accurately retrieved from these models, highlighting the presence of memorization.

Furthermore, we demonstrate that LLMs reflect the dataset’s inherent popularity bias and that their performance as RSs is closely tied to this memorization. This study provides the first evidence that MovieLens-1M has been incorporated into the training of Large Language Models (LLMs), raising potential concerns about the validity of current evaluation practices for LLM-based recommenders.

Future works will be on developing ML-optimized memorization extraction techniques [27] as well as mitigation approaches [20] to reduce memorization and enhance the reliability of LLM evaluation and usage in recommendation tasks.

Acknowledgements. This study was carried out within the MOST – Sustainable Mobility National Research Center funded by the European Union Next- GenerationEU (Italian National Recovery and Resilience Plan (NRRP) – M4C2, Investment 1.4 – D.D. 1033 17/06/2022, CN00000023 – CUP: D93C22000410001). We acknowledge ISCRA for awarding this project access to the LEONARDO supercomputer, hosted by CINECA (Italy).

References

- [1] Himan Abdollahpour, Masoud Mansoury, Robin Burke, Bamshad Mobasher, and Edward C. Malthouse. 2021. User-centered Evaluation of Popularity Bias in Recommender Systems. In *UMAP*. ACM, 119–129.
- [2] Ali Al-Kaswan, Maliheh Izadi, and Arie van Deursen. 2024. Traces of Memorisation in Large Language Models for Code. In *ICSE*. ACM, 78:1–78:12.
- [3] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation. In *RecSys*. ACM, 1007–1014.
- [4] Giovanni Maria Biancofiore, Dario Di Palma, Claudio Pomo, Fedelucio Narducci, and Tommaso Di Noia. 2025. *Conversational User Interfaces and Agents*. Springer Nature Switzerland, Cham, 399–438. https://doi.org/10.1007/978-3-031-61375-3_4
- [5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, and Jared Kaplan et al. 2020. Language Models are Few-Shot Learners. In *NeurIPS*.
- [6] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramèr, and Chiyuan Zhang. 2023. Quantifying Memorization Across Neural Language Models. In *ICLR*. OpenReview.net.
- [7] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting Training Data from Large Language Models. In *USENIX Security Symposium*. USENIX Association, 2633–2650.
- [8] Oscar Celma and Perfecto Herrera. 2008. A new approach to evaluating novel recommendations. In *RecSys*. ACM, 179–186.
- [9] Paolo Cremonesi, Roberto Turrin, Eugenio Lentini, and Matteo Matteucci. 2008. An Evaluation Methodology for Collaborative Recommender Systems. In *International Conference on Automated Solutions for Cross Media Content and Multi-Channel Distribution*. 224–231. <https://doi.org/10.1109/AXMEDIS.2008.13>
- [10] 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 *RecSys*. ACM, 1126–1132.
- [11] Dario Di Palma. 2023. Retrieval-augmented Recommender System: Enhancing Recommender Systems with Large Language Models. In *RecSys*. ACM, 1369–1373.
- [12] Dario Di Palma, Giovanni Maria Biancofiore, Vito Walter Anelli, Fedelucio Narducci, Tommaso Di Noia, and Eugenio Di Sciascio. 2023. Evaluating ChatGPT as a Recommender System: A Rigorous Approach. *CoRR* abs/2309.03613 (2023).
- [13] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, and et al. 2024. The Llama 3 Herd of Models. *CoRR* abs/2407.21783 (2024).
- [14] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2021. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *CoRR* abs/2101.00027 (2021).
- [15] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with Language Model is Planning with World Model. In *EMNLP*. Association for Computational Linguistics, 8154–8173.
- [16] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR*. ACM, 639–648.
- [17] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian J. McAuley, and Wayne Xin Zhao. 2024. Large Language Models are Zero-Shot Rankers for Recommender Systems. In *ECIR (2) (Lecture Notes in Computer Science, Vol. 14609)*. Springer, 364–381.
- [18] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. LLaRA: Large Language-Recommendation Assistant. In *SIGIR*. ACM, 1785–1795.
- [19] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Comput.* 7, 1 (2003), 76–80.
- [20] Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Xiaojun Xu, Yuguang Yao, Hang Li, Kush R. Varshney, Mohit Bansal, Sanmi Koyejo, and Yang Liu. 2024. Rethinking Machine Unlearning for Large Language Models. *CoRR* abs/2402.08787 (2024).
- [21] Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Christopher Leung, Jiajie Tang, and Jiebo Luo. 2024. LLM-Rec: Personalized Recommendation via Prompting Large Language Models. In *NAACL-HLT (Findings)*. Association for Computational Linguistics, 583–612.
- [22] Tommaso Di Noia, Pasquale Lops, Thorsten Joachims, Katrien Verbert, Pablo Castells, Zhenhua Dong, and Ben London (Eds.). 2024. *Proceedings of the 18th ACM Conference on Recommender Systems, RecSys 2024, Bari, Italy, October 14-18, 2024*. ACM.
- [23] OpenAI. 2023. GPT-4 Technical Report. *CoRR* abs/2303.08774 (2023).
- [24] 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.
- [25] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI AUAI Press*, 452–461.
- [26] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *CSCW*. ACM, 175–186.
- [27] Lennart Schneider, Martin Wistuba, Aaron Klein, Jacek Golebiowski, Giovanni Zappella, and Felice Antonio Merra. 2024. Hyperband-based Bayesian Optimization for Black-box Prompt Selection. *CoRR* abs/2412.07820 (2024).
- [28] Tianhao Shi, Yang Zhang, Zhijian Xu, Chong Chen, Fuli Feng, Xiangnan He, and Qi Tian. 2024. Preliminary Study on Incremental Learning for Large Language Model-based Recommender Systems. In *CIKM*. ACM, 4051–4055.
- [29] Harald Steck. 2019. Embarrassingly Shallow Autoencoders for Sparse Data. In *WWW*. ACM, New York, NY, USA, 3251–3257.
- [30] Zhu Sun, Hongyang Liu, Xinghua Qu, Kaidong Feng, Yan Wang, and Yew Soon Ong. 2024. Large Language Models for Intent-Driven Session Recommendations. In *SIGIR*. ACM, 324–334.
- [31] Changxin Tian, Binbin Hu, Chunjing Gan, Haoyu Chen, Zhuo Zhang, Li Yu, Ziqi Liu, Zhiqiang Zhang, Jun Zhou, and Jiawei Chen. 2024. ReLand: Integrating Large Language Models’ Insights into Industrial Recommenders via a Controllable Reasoning Pool. In *RecSys*. ACM, 63–73.
- [32] Tobias Vente, Lukas Wegmeth, Alan Said, and Joeran Beel. 2024. From Clicks to Carbon: The Environmental Toll of Recommender Systems. In *Proceedings of the 18th ACM Conference on Recommender Systems (RecSys ’24)* (Bari, Italy). ACM, New York, NY, USA, 1–11. <https://doi.org/10.1145/3640457.3688074>
- [33] Hangyu Wang, Jianghao Lin, Xiangyang Li, Bo Chen, Chenxu Zhu, Ruiming Tang, Weinan Zhang, and Yong Yu. 2024. FLIP: Fine-grained Alignment between ID-based Models and Pretrained Language Models for CTR Prediction. In *RecSys*. ACM, 94–104.
- [34] Jianling Wang, Haokai Lu, James Caverlee, Ed H. Chi, and Minmin Chen. 2024. Large Language Models as Data Augmenters for Cold-Start Item Recommendation. In *WWW (Companion Volume)*. ACM, 726–729.
- [35] Yan Wang, Zhixuan Chu, Xin Ouyang, Simeng Wang, Hongyan Hao, Yue Shen, Jinjie Gu, Siqiao Xue, James Zhang, Qing Cui, Longfei Li, Jun Zhou, and Sheng Li. 2024. LLMRG: Improving Recommendations through Large Language Model Reasoning Graphs. In *AAAI*. AAAI Press, 19189–19196.
- [36] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *NeurIPS*.
- [37] Wei Wei, Xubin Ren, Jiabin Tang, Qingyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. LLMRec: Large Language Models with Graph Augmentation for Recommendation. In *WSDM*. ACM, 806–815.
- [38] Yunxia Xi, Weiwen Liu, Jianghao Lin, Xiaoling Cai, Hong Zhu, Jieming Zhu, Bo Chen, Ruiming Tang, Weinan Zhang, and Yong Yu. 2024. Towards Open-World Recommendation with Knowledge Augmentation from Large Language Models. In *RecSys*. ACM, 12–22.
- [39] Grace Hui Yang, Hongning Wang, Sam Han, Claudia Hauff, Guido Zucco, and Yi Zhang (Eds.). 2024. *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*. ACM.
- [40] Shenghao Yang, Weizhi Ma, Peijie Sun, Qingyao Ai, Yiqun Liu, Mingchen Cai, and Min Zhang. 2024. Sequential Recommendation with Latent Relations based on Large Language Model. In *SIGIR*. ACM, 335–344.
- [41] Weichen Yu, Tianyu Pang, Qian Liu, Chao Du, Bingyi Kang, Yan Huang, Min Lin, and Shuicheng Yan. 2023. Bag of Tricks for Training Data Extraction from Language Models. In *ICML (Proceedings of Machine Learning Research, Vol. 202)*. PMLR, 40306–40320.
- [42] Eva Zangerle and Christine Bauer. 2023. Evaluating Recommender Systems: Survey and Framework. *ACM Comput. Surv.* 55, 8 (2023), 170:1–170:38.
- [43] Jie Zhang, Li Chen, Shlomo Berkovsky, Min Zhang, Tommaso Di Noia, Justin Basilico, Luiz Pizzato, and Yang Song (Eds.). 2023. *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023, Singapore, Singapore, September 18-22, 2023*. ACM.
- [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. In *I (Still) Can’t Believe It’s Not Better! NeurIPS 2021 Workshop*. <https://openreview.net/forum?id=hFx3fY7-m9b>
- [45] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large Language Models are Human-Level Prompt Engineers. In *ICLR*. OpenReview.net.
- [46] Yaochen Zhu, Liang Wu, Qi Guo, Liangjie Hong, and Jundong Li. 2024. Collaborative Large Language Model for Recommender Systems. In *WWW*. ACM, 3162–3172.