Can Large Language Models Mitigate Popularity Bias in Personalized Recommender Systems?

Md Aminul Islam
University of Illinois Chicago
mislam34@uic.edu

Md Mezbaur Rahman
University of Illinois Chicago
mrahma56@uic.edu

Mehrab Mustafy Rahman
University of Illinois Chicago
mrahm@uic.edu

Abstract

Recommender systems have become essential to modern online platforms, guiding users toward personalized content across a wide range of domains. These systems typically infer user preferences from interaction data (e.g., clicks), which is inherently biased towards popular items. Popularity bias reinforces the frequent exposure of popular items while underrecommending relevant but less popular ones, causing models trained on such data to misrepresent true user preferences. With the advent of large language models (LLMs), recent work has begun to explore their potential as flexible recommenders that can reason over user histories, item attributes, and natural-language instructions. Yet, the role of popularity bias in LLM-based recommendation remains relatively underexplored. In this paper, we study LLMs as popularity-aware ranker for personalized recommender systems. We propose a two-stage framework in which a traditional recommender first retrieves candidate items, and an LLM then re-ranks them using user histories, item attributes, candidate information, and explicit popularity statistics. We evaluate Simple prompting, Chain-of-Thought (CoT) prompting, and ReAct-style prompting on MovieLens using ranking accuracy, average item popularity, and exposure inequality. Our results with DeepSeek and Llama show that Simple prompting achieves the strongest overall NDCG, while structured popularity-aware prompting can reduce popularity bias and improve diversity in a model-dependent manner. In particular, ReAct substantially reduces Avg-Pop and Gini for DeepSeek, whereas CoT provides stronger diversity gains for Llama. These findings highlight both the promise and limitations of LLM-based popularity-aware recommender systems, showing that prompt design can meaningfully shape the accuracy–diversity trade-off in recommender systems. Our code is publicly available at https://github.com/aminul7506/RecLLMPopBiasCorrection.

1 Introduction

Recommender systems have become an essential component of modern online platforms, helping users navigate large collections of items and discover content aligned with their interests. They are widely used in domains such as e-commerce, entertainment, search, social media, video-sharing platforms, and music services (Chen et al., 2023). Most recommender systems learn user preferences from historical interaction data, such as clicks, ratings, views, purchases, or listening records, and use these signals to generate personalized ranked lists (Wu et al., 2010). Although such feedback is easy to collect at scale (Saito et al., 2020) and provides useful behavioral signals (Sanderson, 2010), observed interactions do not necessarily reflect users' true preferences. Instead, they are shaped by exposure mechanisms, including which items are shown, how frequently they appear, and where they are placed in the ranked list (Oosterhuis and de Rijke, 2020; Ovaisi et al., 2020; Ai et al., 2018; Joachims et al., 2017; Luo et al., 2023). Consequently, models trained directly on interaction data may learn biased relevance signals rather than purely personalized preferences (Joachims et al., 2017; Yue et al., 2010).

One of the most persistent forms of bias in recommender systems is popularity bias, where a system disproportionately recommends items that are already popular while giving limited exposure to less popular but potentially relevant items (Yao and Huang, 2017; Cañamares and Castells, 2018; Zhang et al., 2022). This problem is closely related to the long-tailed structure of user–item interaction data, in which a small fraction of head items receives a large share of observed feedback while the majority of tail items remains sparsely interacted with (Zhang et al., 2022). As a result, recommendation models may use popularity as a shortcut for relevance, overestimating head items and underes-

timating niche items that better match individual users' interests (Boratto et al., 2021). Through repeated recommendation and user feedback, this effect can be amplified over time, creating a feedback loop in which popular items receive more exposure, more interactions, and even higher future recommendation scores (Chaney et al., 2018; Klimashevskaia et al., 2024).

Several methods have been proposed to mitigate popularity bias in recommender systems, including inverse propensity weighting (IPW)-based reweighting (Gruson et al., 2019; Joachims et al., 2017; Ai et al., 2018), regularization methods that penalize dependence between scores and popularity (Boratto et al., 2021; Wasilewski and Hurley, 2016; Chen et al., 2020), causal approaches that separate true preference from popularity-driven effects (Wei et al., 2021; Bonner and Vasile, 2018; Wang et al., 2021; Ning et al., 2024), and post-processing or post-hoc correction methods (Zhu et al., 2021; Chen et al., 2024; Islam et al., 2026). Meanwhile, GNN-based recommendation models such as LightGCN (He et al., 2020) and SGL (Wu et al., 2021) achieve strong performance by propagating information over the user–item graph, but this propagation can also amplify popularity bias (Zhou et al., 2023; Chen et al., 2024). Because popular items are connected to many users, they contribute more frequently during message propagation, making their representations overly similar with many users (Zhou et al., 2023).

Conventional recommender systems are often task-specific, relying on separate datasets, features, and architectures, which limits their scalability and transferability. Recent studies therefore explore large language models (LLMs) as flexible recommendation backbones due to their strong language understanding and ability to reason over textual descriptions, item attributes, user histories, and natural-language instructions (Zhou et al., 2025; Liu et al., 2023; Kusano et al., 2024, 2025). By converting interaction histories into structured prompts (Zhang et al., 2021), LLMs can leverage semantic information often unavailable to traditional models and support unified recommendation frameworks such as P5 (Geng et al., 2022) and M6-Rec (Cui et al., 2022). Recent LLMs further enable zero-shot and few-shot recommendation by inferring user preferences and ranking items directly from prompts (Liu et al., 2023). However, LLMs may also inherit popularity bias. Since popular items are more frequently discussed in web text, reviews, and other large-scale corpora used during pretraining, LLMs may favor well-known items when generating recommendations (Ortega et al., 2024; Deldjoo, 2025). Prior work has begun to examine bias in LLM-based recommendation, but there is still limited understanding of whether structured prompts can actively mitigate popularity bias rather than merely reproduce it (Ortega et al., 2024; Deldjoo, 2025). This gap motivates the central question of this paper: can LLMs serve as bias-aware ranking model that reduce popularity bias while preserving personalized recommendation quality?

In this paper, we investigate whether prompt-based LLMs can mitigate popularity bias by re-ranking candidate items generated by traditional recommendation models. Instead of asking an LLM to recommend from the vast set of item catalogs, we first use a conventional recommendation model, such as LightGCN or SGL, to retrieve a top-k candidate set for each user. Then, an LLM re-ranks these candidates using the user's interaction history, item attributes, candidate item information, and item- and dataset-level popularity statistics. This two-stage design leverages traditional recommendation models for initial candidate generation and LLMs for semantically informed re-ranking, while explicitly considering dataset- and item-specific popularity bias. To guide the re-ranking process, we study structured prompting strategies that explicitly encourage the LLM to reason about user preferences and popularity information. In particular, we consider Zero-Shot Chain-of-Thought (CoT) prompting (Wei et al., 2022b), where the model is instructed to analyze the user's historical preferences, compare candidate items based on semantic and feature-level similarity, and account for item popularity before producing the final ranking. We also consider ReAct-style prompting (Yao et al., 2023a), where the model alternates between reasoning about user preferences and taking ranking actions. In this framework, the LLM first extracts a user taste profile from previously interacted items and then re-ranks the candidate set by considering preference match, item features, and popularity information.

This work studies the potential of LLMs as popularity-aware re-rankers for personalized recommendation. Our main contributions are as follows:

- We formulate the problem of using prompt-based

LLMs as popularity-aware re-rankers that operate on candidate items generated by traditional recommendation models.

- We propose a two-stage recommendation framework that first retrieves top-k candidates using conventional models, and then applies LLM-based re-ranking using user histories, item attributes, candidate information, and popularity statistics.

- We design and evaluate structured prompting strategies, including Zero-Shot CoT and ReAct-style prompting, to encourage LLMs to jointly reason about user preference matching, item semantics, and popularity information. To the best of our knowledge, this is the first work to systematically design and evaluate CoT- and ReAct-style prompting strategies for popularity-aware LLM-based re-ranking.

- We provide an empirical framework for analyzing whether LLM-based re-ranking can improve personalized recommendation quality and increase diversity while reducing the influence of global item popularity.

2 Related Work

Recommender systems often suffer from popularity bias, where frequently interacted items receive disproportionate exposure and dominate recommendation results (Chen et al., 2022; Zhang et al., 2022; Chen et al., 2020; Rhee et al., 2022).

Traditional recommender systems. Existing traditional popularity debiasing methods mainly fall into four categories: IPW, regularization, post-processing re-ranking, and causal inference. IPW-based methods down-weight interactions involving popular items by assigning weights inversely proportional to item popularity (Ai et al., 2018; Joachims et al., 2017; Zhou et al., 2023; Kim et al., 2022), but they often suffer from high variance and depend heavily on accurate propensity estimation (Gruson et al., 2019; Bottou et al., 2013; Ovaisi et al., 2021). Regularization-based methods reduce popularity effects during training by imposing diversity, mean-matching, knowledge-transfer, or decorrelation constraints (Wasilewski and Hurley, 2016; Kamishima et al., 2014; Chen et al., 2020; Boratto et al., 2021). Post-processing methods instead mitigate popularity bias after training by adjusting recommendation lists or modifying learned node representations. These methods improve objectives such as calibration, diversity, and popularity fairness, or remove popularity-related components from user and item representations (Steck, 2018; Abdollahpouri et al., 2019; Zhu et al., 2021; Chen et al., 2024; Islam et al., 2026). Causal inference methods explicitly model popularity as a causal or confounding factor and aim to separate it from true user preference (Bonner and Vasile, 2018; Wang et al., 2021; Wei et al., 2021; Ning et al., 2024; He et al., 2022; Zhao et al., 2022). Examples include CausE (Bonner and Vasile, 2018), MPCI (He et al., 2022), MACR (Wei et al., 2021), and PPAC (Ning et al., 2024).

LLM-based recommender systems. LLMs have recently gained increasing attention in recommender systems due to their strong reasoning and language understanding capabilities (Liu et al., 2023). Recent surveys characterize LLM-based recommendation as a decision-making process in which the model directly produces recommendations, commonly using pointwise, pairwise, or listwise strategies (Wu et al., 2024). Pointwise methods evaluate each candidate item independently given the user's interaction history (Zheng et al., 2024). For example, TALLRec formulates recommendation as binary classification (Bao et al., 2023), while E4SRec uses an adapter-based framework to estimate item-level recommendation probabilities (Liu et al., 2023). Pairwise methods instead ask the LLM to compare two candidate items and select the more relevant one, with approaches such as the sliding-window ranking strategy proposed by (Qin et al., 2023). LLM-based recommender systems may also suffer from popularity bias because popular items are more likely to appear frequently in pre-training corpora, user histories, and prompt contexts. Recent studies show that LLM recommenders can over-recommend well-known items and that their outputs are sensitive to prompt design (Lichtenberg et al., 2024; Deldjoo, 2025). To mitigate this issue, recent work explores prompt-based debiasing and popularity-aware prompting strategies. For example, FairLRM (Luo et al., 2026) uses LLMs to separate semantic user preference from global popularity effects. Other prompt-engineering approaches also aim to balance recommendation accuracy with exposure fairness (Hamad, 2025; Das and Sakib, 2024).

3 Problem Description

Recommender systems often suffer from popularity bias, where highly popular items receive disproportionate exposure while less popular but potentially relevant long-tail items are under-recommended. This issue is especially problematic because recommendation feedback loops can further amplify item popularity over time, i.e., popular items are recommended more frequently, receive more user interactions, and consequently become even more likely to be recommended in the future.

In this work, we study whether LLMs can mitigate popularity bias in personalized recommender systems. Although LLMs have strong semantic reasoning capabilities, they may also inherit popularity bias from their pretraining data, where globally popular items are more frequently represented. Therefore, our goal is to investigate whether carefully designed prompts can guide LLMs to generate recommendations that better align with individual user preferences rather than relying primarily on globally popular items.

Formally, given a user u, their historical interactions $\mathcal{H}_u$, a candidate item set $\mathcal{C}_u$, item features $\mathcal{X}$, and item popularity information $\mathcal{P}$, the goal is to generate a ranked list of k items:

$$\pi_u = [i_1, i_2, \ldots, i_k], \quad i_j \in \mathcal{C}_u, \tag{1}$$

where the ranking should be both relevant to the user's personal preferences and less dominated by globally popular items.

Let π_u^* denote an ideal unbiased ranking that reflects the user's true preferences while avoiding excessive popularity-driven exposure. The objective is to select a ranking π_u that minimizes the discrepancy from this unbiased ranking:

$$\pi_u^\star = \arg\min_{\pi_u \in \Pi_u} d(\pi_u, \pi_u^*), \tag{2}$$

where Π_u is the set of possible candidate rankings and $d(\cdot)$ measures the difference between the generated ranking and the unbiased target ranking. Since π_u^* is not directly observable in practice, we evaluate the generated ranking using both ranking accuracy metrics and popularity-bias metrics. s

4 Methodology

We propose a two-stage recommendation framework that combines a traditional recommender model with an LLM-based re-ranking module. The main motivation is that conventional recommender models are effective at retrieving relevant candidates from large item catalogs, while LLMs can use semantic reasoning over user history, item attributes, and popularity information to produce a more preference-aware and bias-aware final ranking.

4.1 Stage 1: Candidate generation

In the first stage, we train a traditional recommendation model (e.g., LightGCN (He et al., 2020)) using user-item interaction data. Given a user u, the recommender systems estimates a relevance score $s(u, i)$ for each item $i \in \mathcal{I}$, where $\mathcal{I}$ denotes the full item catalog. The model then retrieves a top-K candidate set:

$$\mathcal{C}_u = \mathrm{TopK}_{i \in \mathcal{I}}\, s(u, i). \tag{3}$$

Instead of prompting the LLM with the entire item catalog, we provide only the top-K candidate items. This reduces prompt length, lowers computational cost, and allows the LLM to focus on re-ranking a smaller set of potentially relevant items.

4.2 Stage 2: LLM-based popularity-aware re-ranking

In the second stage, we use an LLM to rerank the candidate items generated by the traditional recommender. The LLM receives four types of information: the user's historical interactions, item metadata, item-level popularity scores, and dataset-level popularity statistics such as entropy, KL-divergence, and Gini index.

The core idea is to explicitly instruct the LLM to reason about both user preference alignment and popularity bias. Rather than directly selecting globally popular items, the LLM is prompted to identify recurring patterns in the user's history and compare each candidate item against those preferences. For example, in a movie recommendation setting, the LLM may reason about genre, theme, tone, release year, and popularity before producing the final ranked list.

We use prompts that encourage the model to follow the ranking objective:

$$S_{\text{LLM}}(u, i) = \alpha \cdot \text{PrefMatch}(u, i) + \gamma \cdot \text{Novelty}(i) - \lambda \cdot \text{Popularity}(i). \tag{4}$$

where $\text{PrefMatch}(u, i)$ measures how well item i matches the inferred preference profile of user u, $\text{Novelty}(i)$ encourages relevant long-tail or less

obvious items, and Popularity(i) penalizes over-reliance on globally popular items. The LLM does not explicitly optimize this equation, but the prompt is designed to guide the model toward this decision logic.

4.3 Prompting strategies

We consider two prompting strategies for LLM-based re-ranking.

Zero-Shot Chain-of-Thought prompting. We first use zero-shot CoT prompting (Wei et al., 2022a) to encourage the LLM to reason step-by-step before generating the final recommendation list. The prompt asks the model to analyze the user's historical interactions, infer their preferences, compare candidate items based on item features, and consider popularity information before producing the final ranking. This encourages the LLM to base its recommendations on personalized preference signals rather than simply selecting highly popular items.

ReAct-style prompting. We also consider a ReAct-style prompting (Yao et al., 2023b) strategy, where the LLM alternates between reasoning and ranking actions. The model first extracts a user preference profile from the interaction history, then compares each candidate item with this profile, and finally updates the ranking based on relevance, novelty, and popularity penalty. This structured prompting strategy makes the re-ranking process more interpretable and forces the model to explicitly connect its reasoning to the final recommendation output. Example CoT and ReAct-style prompts are shown in Figure 1 in the Appendix.

4.4 Novelty and motivation

The novelty of our approach lies in using the LLM not as a standalone recommender, but as a popularity-aware semantic reranker on top of a traditional recommendation model. This design has several advantages.

First, the traditional recommender efficiently narrows the search space to a relevant candidate set, avoiding the need to prompt the LLM with the full item catalog. Second, the LLM can incorporate semantic information from item metadata, such as genre, theme, style, and description, which may not be fully captured by collaborative filtering models. Third, by explicitly providing item popularity and dataset-level imbalance statistics, the LLM is encouraged to distinguish between items that are genuinely relevant to the user and items that are recommended primarily because they are globally popular.

Overall, our method is well-suited for popularity bias mitigation because the problem is not only a ranking problem, but also a reasoning problem. A recommendation system should understand why an item is being recommended and whether it reflects the user's actual preferences. By combining traditional candidate generation with LLM-based popularity-aware re-ranking, our approach aims to preserve recommendation relevance while reducing popularity-driven exposure.

5 Experimental Setup

We aim to evaluate our method by answering the following research questions::

- **RQ1:** How does our method affect personalized recommendation accuracy compared with standard prompt-based recommendation method?
- **RQ2:** Can our method reduce popularity bias in LLM-based recommendations, as measured by the average popularity of recommended items?
- **RQ3:** Does our method improve recommendation diversity by producing more balanced item recommendations across the catalog?
- **RQ4:** How does our method affect recommendation performance for popular and niche (long-tail) item groups, and does improving niche-item recommendation come at the cost of popular-item performance?

5.1 Dataset

We conduct our experiments on the MovieLens-100K dataset[1], a widely used benchmark for recommender system evaluation. The dataset contains 100,000 user–movie ratings collected from 943 users and 1,682 movies. Ratings are given on a discrete scale from 1 to 5. In addition to the rating records, MovieLens-100K provides user-side demographic information, including age, gender, occupation, and zip code, as well as movie-side metadata such as movie titles and genre labels. The dataset is distributed across multiple files, including the main interaction file containing user IDs, movie IDs, ratings, and timestamps, together with separate files describing user and movie attributes. We preprocess the data to make it compatible with the

[1] https://grouplens.org/datasets/movielens/100k

recommendation models and then post-process the resulting user histories and candidate items into the textual format required for LLM prompting. For each user, we split the interactions into 70% for training, 10% for validation, and 20% for testing. Details of the dataset is shown in Table 1.

5.2 Evaluation metrics

We evaluate recommendation performance using Normalized Discounted Cumulative Gain (NDCG), a widely used top-k ranking metric that considers both the relevance of recommended items and their positions in the ranked list. k denotes the cutoff position of the recommendation list. To assess popularity bias, we also report the average popularity of recommended items, denoted as AvgPop@k. This metric measures the average popularity of the items appearing in the top-k recommendation list and allows us to examine whether a method reduces the tendency to recommend globally popular items. Lower AvgPop@k indicates that the recommendation list contains less popular items on average. In addition, we use the Gini index to measure recommendation diversity at the top-k level. The Gini index captures how unevenly recommendations are distributed across items: a higher Gini value indicates that recommendations are concentrated on a small set of items, while a lower Gini value suggests a more diverse distribution across the item catalog. We report all metrics for $k = 10$ and $k = 20$.

5.3 Baselines

We compare our method with a prompt-based LLM recommendation baseline that generates item recommendations without explicit popularity-aware instructions. This comparison allows us to examine whether incorporating popularity-aware prompting improves recommendation accuracy, reduces popularity bias, and increases recommendation diversity in LLM-based recommendation.

5.4 Implementation details

We adopt GNN-based LightGCN (He et al., 2020) as the backbone recommender to generate initial candidate items for each user. For LightGCN, we rely on the publicly available PyTorch implementation released by the original authors. The learning rate is selected from $\{10^{-5}, 10^{-4}, 10^{-3}\}$, the batch size is set to 2,048, the embedding dimension to 256, and we use a two-layer graph convolutional architecture with all remaining hyperparameters kept at their defaults. Since LightGCN operates on implicit interactions, we convert MovieLens ratings into a binary user–item graph, treating ratings of 3 or higher as positive interactions and all others as 0. For the LLM re-ranking stage, we employ DeepSeek-V3 (DeepSeek Chat), accessed via the DeepSeek API, and Llama 3.1 8B (Grattafiori et al., 2024), served locally using vLLM (Kwon et al., 2023). Both models are used in inference-only mode with temperature set to 0 for deterministic outputs.

Table 1: Statistics of the MovieLens-100K dataset.

Dataset	MovieLens-100K
#Users	943
#Items	1,682
#Ratings	100,000
Rating scale	1–5
User metadata	Age, gender, occupation
Item metadata	Movie title, genre
Interaction fields	User ID, movie ID, rating
Train / Val. / Test	70% / 10% / 20%

6 Results

RQ1: Recommendation accuracy. Table 2 shows that Simple prompting achieves the strongest overall ranking accuracy for both models. For DeepSeek, CoT is second-best, while ReAct causes a larger drop, suggesting that the explicit reasoning-action process may move the ranking away from the original relevance signal. For Llama, ReAct remains close to Simple and outperforms CoT, indicating that Llama is more robust to ReAct-style prompting. Overall, popularity-aware prompting does not improve NDCG, and may introduce an accuracy–diversity trade-off.

RQ2: Popularity-bias mitigation. Table 3 reports AvgPop, where lower values indicate less popularity bias. For DeepSeek, ReAct substantially reduces AvgPop, showing that structured popularity-aware reasoning can push recommendations away from globally popular items. CoT provides only a small reduction. For Llama, CoT gives the lowest AvgPop, while ReAct increases popularity. Thus, popularity-aware prompting can reduce popularity bias, but its effect depends strongly on the interaction between the model and prompting strategy.

RQ3: Recommendation diversity. Table 4 shows exposure inequality using Gini, where lower

Table 2: Ranking performance on MovieLens. Higher NDCG indicates better recommendation accuracy. The best-performing method is shown in bold and the second-best is underlined.

Prompting	DeepSeek		Llama	
	NDCG@10	NDCG@20	NDCG@10	NDCG@20
Simple	**0.2746**	**0.2952**	**0.2729**	**0.3007**
CoT	<u>0.2659</u>	<u>0.2912</u>	0.2580	0.2864
ReAct	0.2328	0.2608	<u>0.2680</u>	<u>0.2979</u>

Table 3: Average popularity of recommended items on MovieLens. Lower AvgPop indicates less popularity bias. The best-performing method is shown in bold and the second-best is underlined.

Prompting	DeepSeek		Llama	
	AvgPop@10	AvgPop@20	AvgPop@10	AvgPop@20
Simple	156.3091	148.5783	<u>161.4948</u>	<u>157.2458</u>
CoT	<u>155.4800</u>	<u>147.0497</u>	**157.8766**	**156.4410**
ReAct	**125.7753**	**126.8206**	163.8947	161.4252

values indicate more balanced item exposure. The trends mostly match AvgPop. For DeepSeek, ReAct gives the largest diversity gain, reducing both Gini@10 and Gini@20. CoT gives only minor improvements. For Llama, CoT achieves the best Gini scores, while ReAct increases exposure concentration. These results suggest that popularity-aware prompting can improve diversity, but the benefit is model-dependent.

RQ4: Popular vs. niche item analysis. In response to RQ4, we investigate the performance across popular and niche item groups and analyzes whether gains in one group come at the expense of the other. Following (Abdollahpouri et al., 2019), we sort items by interaction frequency and define the smallest set of items accounting for 80% of interactions as popular items, while the remaining items are treated as long-tail items. Tables 5–6 analyze performance on popular and niche item groups. For popular items, Simple prompting remains best for NDCG, while ReAct reduces AvgPop and Gini most effectively for DeepSeek and CoT does so for Llama. For niche items, reasoning-based prompts often improve accuracy: CoT leads DeepSeek on NDCG@10, ReAct leads DeepSeek on NDCG@20, and ReAct performs best for Llama. Diversity changes are smaller in the niche subset because these items already come from the lower-popularity region. Overall, the trade-off between accuracy, popularity bias, and diversity is model- and metric-dependent.

7 Conclusion and Future Work

In this paper, we investigate whether prompt-based LLMs can serve as popularity-aware re-rankers for personalized recommendation. We propose a two-stage framework in which a traditional recommendation model first retrieves candidate items, and an LLM then re-ranks those candidates using user histories, item attributes, candidate information, and popularity statistics. We evaluate Simple, Chain-of-Thought, and ReAct-style prompting strategies on MovieLens using NDCG for ranking accuracy, AvgPop for popularity bias, and Gini for exposure inequality.

Our results show that LLM-based re-ranking introduces a clear accuracy–diversity trade-off. Simple prompting achieves the strongest overall NDCG for both DeepSeek and Llama, suggesting that less constrained prompts better preserve the original

Table 4: Recommendation exposure inequality on MovieLens. Lower Gini indicates more equal and diverse item recommendation. The best-performing method is shown in bold and the second-best is underlined.

Prompting	DeepSeek		Llama	
	Gini@10	Gini@20	Gini@10	Gini@20
Simple	0.8634	0.8293	<u>0.8490</u>	<u>0.8286</u>
CoT	<u>0.8605</u>	<u>0.8271</u>	**0.8419**	**0.8267**
ReAct	**0.7856**	**0.7746**	0.8528	0.8366

Table 5: NDCG@10 and NDCG@20 on popular-item and niche-item groups. Predictions are filtered to the corresponding item group before scoring. Higher NDCG indicates better recommendation accuracy. The best-performing method is shown in **bold** and the second-best is underlined.

Prompting	Popular Items				Niche Items			
	DeepSeek		Llama		DeepSeek		Llama	
	NDCG@10	NDCG@20	NDCG@10	NDCG@20	NDCG@10	NDCG@20	NDCG@10	NDCG@20
Simple	**0.2911**	**0.3231**	**0.2893**	**0.3300**	0.1313	0.1463	0.1339	0.1484
CoT	0.2799	0.3172	0.2727	0.3149	**0.1338**	0.1475	0.1346	0.1480
ReAct	0.2532	0.2935	0.2818	0.3248	0.1321	**0.1484**	**0.1407**	**0.1529**

Table 6: AvgPop and Gini on the popular-item and niche-item groups. Lower AvgPop indicates less popularity bias, and lower Gini indicates more equal item exposure. The best-performing method is shown in **bold** and the second-best is underlined.

Group	Prompting	DeepSeek				Llama			
		AvgPop@10	AvgPop@20	Gini@10	Gini@20	AvgPop@10	AvgPop@20	Gini@10	Gini@20
Popular	Simple	161.0040	155.7750	0.6335	0.5634	167.3536	163.6044	0.6050	0.5586
	CoT	160.6978	154.2981	0.6312	0.5585	**163.9436**	**162.8149**	**0.5885**	**0.5533**
	ReAct	**133.3799**	**137.1474**	**0.4863**	**0.4583**	168.8339	166.7897	0.6087	0.5718
Niche	Simple	31.6399	31.0567	0.7601	0.7376	31.6262	31.0627	0.7596	0.7378
	CoT	31.6052	**31.0450**	0.7583	**0.7370**	**31.6132**	**31.0592**	**0.7581**	0.7379
	ReAct	**31.5209**	31.0528	**0.7555**	0.7377	31.6473	31.0620	0.7595	**0.7378**

relevance signal. However, structured popularity-aware prompts can reduce popularity bias and improve diversity. For DeepSeek, ReAct substantially lowers AvgPop and Gini, indicating stronger movement away from globally popular items. For Llama, CoT provides the best improvements in AvgPop and Gini, while ReAct is less effective. The popular-versus-niche analysis further shows that reasoning-based prompts can improve niche-item recommendation accuracy in some cases, even when they reduce accuracy on popular items. Overall, the effectiveness of popularity-aware prompting depends strongly on the underlying LLM and the prompting strategy.

There are several directions for future work. First, this study can be extended to additional datasets and domains to test whether the observed model-dependent trends generalize beyond MovieLens. Datasets that contain unbiased test data for evaluation are more helpful for unbiased evaluations. Second, future work should evaluate more LLMs, including both open-source and proprietary models, to better understand how model size, instruction tuning, and reasoning ability affect popularity-aware ranking. Third, more adaptive prompting strategies could be explored, where the prompt dynamically adjusts its emphasis on relevance, diversity, or popularity depending on the user profile or candidate set. Fourth, future methods could combine LLM re-ranking with explicit optimization objectives that jointly balance NDCG, AvgPop, and Gini. Finally, a deeper qualitative analysis of LLM reasoning traces may help explain when popularity-aware prompts successfully identify niche but relevant items and when they over-correct away from useful popular recommendations.

8 Team Work Division and Overall Experience

Md Aminul Islam led the implementation of the LightGCN-based candidate generation pipeline, including model training and integration with the re-ranking stage. Md Mezbaur Rahman was responsible for designing and implementing the ReAct-based prompting strategy. Mehrab Mustafy Rahman developed the Chain-of-Thought prompting approach and its evaluation. The introduction, experimental evaluation, and paper writing were shared equally among all three members.

References

Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2019. Managing popularity bias in recommender systems with personalized re-ranking. In *FLAIRS*.

Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and

W Bruce Croft. 2018. Unbiased learning to rank with unbiased propensity estimation. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 385–394.

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 *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1007–1014.

Stephen Bonner and Flavian Vasile. 2018. Causal embeddings for recommendation. In *Proceedings of the 12th ACM conference on recommender systems*, pages 104–112.

Ludovico Boratto, Gianni Fenu, and Mirko Marras. 2021. Connecting user and item perspectives in popularity debiasing for collaborative recommendation. *Information Processing & Management*, 58(1):102387.

Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. 2013. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research*, 14(1):3207–3260.

Rocío Cañamares and Pablo Castells. 2018. Should i follow the crowd? a probabilistic analysis of the effectiveness of popularity in recommender systems. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 415–424.

Allison JB Chaney, Brandon M Stewart, and Barbara E Engelhardt. 2018. How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. In *Proceedings of the 12th ACM conference on recommender systems*, pages 224–232.

Hao Chen, Zefan Wang, Feiran Huang, Xiao Huang, Yue Xu, Yishi Lin, Peng He, and Zhoujun Li. 2022. Generative adversarial framework for cold-start item recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2565–2571.

Jiajia Chen, Jiancan Wu, Jiawei Chen, Xin Xin, Yong Li, and Xiangnan He. 2024. How graph convolutions amplify popularity bias for recommendation? *Frontiers of Computer Science*, 18(5):185603.

Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2023. Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems*, 41(3):1–39.

Zhihong Chen, Rong Xiao, Chenliang Li, Gangfeng Ye, Haochuan Sun, and Hongbo Deng. 2020. Esam: Discriminative domain adaptation with non-displayed items to improve long-tail performance. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pages 579–588.

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*.

Anindya Bijoy Das and Shahnewaz Karim Sakib. 2024. Unveiling and mitigating bias in large language model recommendations: A path to fairness. *arXiv preprint arXiv:2409.10825*.

Yashar Deldjoo. 2025. Understanding biases in chatgpt-based recommender systems: Provider fairness, temporal stability, and recency. *ACM Transactions on Recommender Systems*, 4(2):1–35.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Alois Gruson, Praveen Chandar, Christophe Charbuillet, James McInerney, Samantha Hansen, Damien Tardieu, and Ben Carterette. 2019. Offline evaluation to make decisions about playlist recommendation algorithms. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 420–428.

Muhammad Hamad. 2025. Decoupling popularity bias and user fairness in llm-based recommendation systems.

Ming He, Changshu Li, Xinlei Hu, Xin Chen, and Jiwen Wang. 2022. Mitigating popularity bias in recommendation via counterfactual inference. In *International Conference on Database Systems for Advanced Applications*, pages 377–388. Springer.

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

Md Aminul Islam, Elena Zheleva, and Ren Wang. 2026. Post-hoc popularity bias correction in gnn-based collaborative filtering. In *Proceedings of the ACM Web Conference 2026*, pages 6818–6829.

Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 781–789.

Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2014. Correcting popularity bias by enhancing recommendation neutrality. *RecSys posters*, 10.

Minseok Kim, Jinoh Oh, Jaeyoung Do, and Sungjin Lee. 2022. Debiasing neighbor aggregation for graph neural network in recommender systems. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4128–4132.

Anastasiia Klimashevskaia, Dietmar Jannach, Mehdi Elahi, and Christoph Trattner. 2024. A survey on popularity bias in recommender systems: A. klimashevskaia et al. *User Modeling and User-Adapted Interaction*, 34(5):1777–1834.

Genki Kusano, Kosuke Akimoto, and Kunihiro Takeoka. 2024. Are longer prompts always better? prompt selection in large language models for recommendation systems. *arXiv preprint arXiv:2412.14454*.

Genki Kusano, Kosuke Akimoto, and Kunihiro Takeoka. 2025. Revisiting prompt engineering: A comprehensive evaluation for llm-based personalized recommendation. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*, pages 832–841.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Jan Malte Lichtenberg, Alexander Buchholz, and Pola Schwöbel. 2024. Large language models as recommender systems: A study of popularity bias. *arXiv preprint arXiv:2406.01285*.

Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. *arXiv preprint arXiv:2304.10149*.

Dan Luo, Lixin Zou, Qingyao Ai, Zhiyu Chen, Dawei Yin, and Brian D Davison. 2023. Model-based unbiased learning to rank. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 895–903.

Renqiang Luo, Dong Zhang, Yupeng Gao, Wen Shi, Mingliang Hou, Jiaying Liu, Zhe Wang, and Shuo Yu. 2026. Bridging semantic understanding and popularity bias with llms. In *Proceedings of the ACM Web Conference 2026*, pages 3656–3665.

Wentao Ning, Reynold Cheng, Xiao Yan, Ben Kao, Nan Huo, Nur Al Hasan Haldar, and Bo Tang. 2024. Debiasing recommendation with personal popularity. In *Proceedings of the ACM Web Conference 2024*, pages 3400–3409.

Harrie Oosterhuis and Maarten de Rijke. 2020. Policy-aware unbiased learning to rank for top-k rankings. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 489–498. Association for Computing Machinery.

Gustavo Mendonça Ortega, Rodrigo Ferrari de Souza, and Marcelo Garcia Manzato. 2024. Evaluating zero-shot large language models recommenders on popularity bias and unfairness: a comparative approach to traditional algorithms. In *Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia)*, pages 45–48. SBC.

Zohreh Ovaisi, Ragib Ahsan, Yifan Zhang, Kathryn Vasilaky, and Elena Zheleva. 2020. Correcting for selection bias in learning-to-rank systems. In *Proceedings of The Web Conference 2020*, pages 1863–1873.

Zohreh Ovaisi, Kathryn Vasilaky, and Elena Zheleva. 2021. Propensity-independent bias recovery in offline learning-to-rank systems. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1763–1767.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, and 1 others. 2023. Large language models are effective text rankers with pairwise ranking prompting. *arXiv preprint arXiv:2306.17563*.

Wondo Rhee, Sung Min Cho, and Bongwon Suh. 2022. Countering popularity bias by regularizing score differences. In *Proceedings of the 16th ACM conference on recommender systems*, pages 145–155.

Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased recommender learning from missing-not-at-random implicit feedback. In *Proceedings of the 13th international conference on web search and data mining*, pages 501–509.

Mark Sanderson. 2010. Test collection based evaluation of information retrieval systems. *Foundations and Trends® in Information Retrieval*, 4(4):247–375.

Harald Steck. 2018. Calibrated recommendations. In *Proceedings of the 12th ACM conference on recommender systems*, pages 154–162.

Wenjie Wang, Fuli Feng, Xiangnan He, Xiang Wang, and Tat-Seng Chua. 2021. Deconfounded recommendation for alleviating bias amplification. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1717–1725.

Jacek Wasilewski and Neil Hurley. 2016. Incorporating diversity in a learning to rank recommender system. In *FLAIRS*, pages 572–578.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022a. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Tianxin Wei, Fuli Feng, Jiawei Chen, Ziwei Wu, Jinfeng Yi, and Xiangnan He. 2021. Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1791–1800.

Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 726–735.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, and 1 others. 2024. A survey on large language models for recommendation. *World Wide Web*, 27(5):60.

Qiang Wu, Christopher JC Burges, Krysta M Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Information Retrieval*, 13:254–270.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023a. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023b. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*.

Sirui Yao and Bert Huang. 2017. Beyond parity: Fairness objectives for collaborative filtering. *Advances in neural information processing systems*, 30.

Yisong Yue, Rajan Patel, and Hein Roehrig. 2010. Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *Proceedings of the 19th international conference on World wide web*, pages 1011–1018.

An Zhang, Wenchang Ma, Xiang Wang, and Tat-Seng Chua. 2022. Incorporating bias-aware margins into contrastive loss for collaborative filtering. *Advances in Neural Information Processing Systems*, 35:7866–7878.

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

Zihao Zhao, Jiawei Chen, Sheng Zhou, Xiangnan He, Xuezhi Cao, Fuzheng Zhang, and Wei Wu. 2022. Popularity bias is not always evil: Disentangling benign and harmful bias for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(10):9920–9931.

Zhi Zheng, Wenshuo Chao, Zhaopeng Qiu, Hengshu Zhu, and Hui Xiong. 2024. Harnessing large language models for text-rich sequential recommendation. In *Proceedings of the ACM Web Conference 2024*, pages 3207–3216.

Bohao Zhou, Yibing Zhan, Zhonghai Wang, Yanhong Li, Chong Zhang, Baosheng Yu, Liang Ding, Hua Jin, Weifeng Liu, Xiongbin Wang, and 1 others. 2025. Benchmarking medical llms on anesthesiology: A comprehensive dataset in chinese. *IEEE Transactions on Emerging Topics in Computational Intelligence*.

Huachi Zhou, Hao Chen, Junnan Dong, Daochen Zha, Chuang Zhou, and Xiao Huang. 2023. Adaptive popularity debiasing aggregator for graph collaborative filtering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 7–17.

Ziwei Zhu, Yun He, Xing Zhao, and James Caverlee. 2021. Popularity bias in dynamic recommendation. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 2439–2449.

Appendix

We provide the sample prompt below for our experiments:

Sample Chain-of-Thought Prompt

Given the user's previously interacted items:

$$\mathcal{H}_u = \{\text{Arrival, Blade Runner 2049, Ex Machina, } \ldots\}$$

and the candidate item set:

$$\mathcal{C}_u = \{\text{Moon, Her, Interstellar, Annihilation, } \ldots\}.$$

Analyze the user's preferences based on item attributes such as genre, theme, style, and release year. Each candidate item is also associated with an item-level popularity score. The dataset has popularity imbalance statistics such as entropy, KL-divergence, and Gini index.
Think step by step about: (i) the user's preference profile, (ii) how well each candidate matches the user's preferences, (iii) whether each candidate is selected due to genuine relevance or global popularity, and (iv) how to balance relevance with reduced popularity bias.
Finally, generate a ranked list of the top-k candidate items from most to least preferred.

Sample ReAct Prompt

You are given a user's interaction history, candidate items, item features, and item popularity information.
Thought 1: Infer the user's preference profile from the previously interacted items. Identify recurring genres, themes, styles, and other content patterns.
Action 1: Summarize the user's preference profile in 3–5 concise traits.
Observation 1: The user appears to prefer slow-burn sci-fi, philosophical themes, psychological tension, and cerebral tone.
Thought 2: Compare each candidate item with the inferred user preference profile while also considering item popularity.
Action 2: Score each candidate using preference match, style match, novelty, and popularity penalty.
Observation 2: Candidates that strongly match the user's preferences and are not selected solely due to high popularity should be ranked higher.
Thought 3: Produce the final recommendation ranking by prioritizing personalized relevance while reducing over-reliance on globally popular items.
Action 3: Return only the final ranked top-k list.

Figure 1: Examples of the prompting strategies used for LLM-based popularity-aware reranking. Chain-of-Thought prompting encourages step-by-step preference and bias reasoning, while ReAct prompting structures the process into reasoning, action, and observation steps.